Bridging network reconstruction and mathematical modelling - rxncon a framework to reconstruct, visualise and model signal-transduction networks

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium
(Dr. rer. nat.)
im Fach Biophysik

eingereicht an der
Lebenswissenschaftlichen Fakultät
der Humboldt-Universität zu Berlin

von MSc. Sebastian Thieme

Präsidentin der Humboldt-Universität zu Berlin
Prof. Dr.-Ing. Dr. Sabine Kunst

Dekan der Lebenswissenschaftlichen Fakultät
Prof. Dr. Bernhard Grimm

Gutachter/innen:	Dr. Marcus Krantz
	Prof. Dr. Uwe Ohler
	Prof. Dr. Heike Siebert
Tag der mündlichen Prüfung:	22.09.2017

Typeset by LaTeX.

Abstract

Living organisms are complex systems of interacting components. A crucial step to understand those complex biological systems is the construction of biological networks that reflect our current knowledge of the system.

The scope and coverage of different network reconstructions can differ, but they have one aim in common – to convert the knowledge into a mathematical model enabling computational analysis to find possible inconsistencies and gaps. While reconstruction methods for metabolic networks are well established, only a few methods exist for reconstructing cellular signal-transduction networks.

In this thesis, I present a method – rxncon – enabling a systematised and condensed reconstruction of signal-transduction networks. This method has two aspects. On the one hand, we developed a language for reconstructing biological networks. The language addresses the issue, that states are combined in signal-transduction networks, which create a large number of specific states, generating highly complex structures. Due to the context-free grammar in the language and the description of the data on the same level of detail as biological findings we can largely avoid the combinatorial complexity. On the other hand, we developed a framework for interpreting and exporting this knowledge into different mathematical models and visualisation formats, enabling a workflow to: 1) reconstruct mechanistic detailed signal-transduction network, 2) convert them into an executable Boolean model for evaluation, validation and improvement of the network and 3) export the reconstructed model into a rule-based model. Hence, rxncon has the potential to reconstruct, validate and simulate large-scale signalling networks – bridging large scale network reconstruction and classical mathematical modelling approaches.

Zusammenfassung

Lebende Organismen sind komplexe Systeme von miteinander interagierenden Komponenten. Ein entscheidender Schritt zum besseren Verständnis solcher biologischen Systeme ist die Erstellung biologischer Netzwerke, welche unser bisheriges Verständnis dieser Systeme widerspiegelt.

Verschiedene Ansätze zur Netzwerk-Rekonstruktion unterscheiden sich zwar in ihrem Zweck und ihrer Komplexität, allerding haben sie ein gemeinsames Ziel: die Übersetzung des biologischen Wissens in ein mathematisches Modell zur Aufdeckung von Inkonsistenzen und Wissenslücken innerhalb der Rekonstruktionen durch computerbasierte Analysen. Während es für metabolische Netzwerke bereits gut entwickelte Rekonstruktionsansätze gibt, existieren derzeit nur wenige Ansätze für Signal-Transduktionsnetzwerke.

In dieser Arbeit stelle ich eine Methode zur systematischen und komprimierten Rekonstruktion von Signal-Transduktionsnetzwerken vor – rxncon. Diese Methode hat zwei grundlegende Aspekte:

Einerseits haben wir eine Sprache zur Rekonstruktion biologischer Netzwerke entwickelt, die die Probleme kombinatorischer Komplexität durch die Kombination von Zuständen während des Rekonstruktionsprozesses angeht. Diese kombinatorische Komplexität wird durch die Verwendung kontextfreier Grammatik und der Beschreibung der Daten auf derselben Ebene wie experimentelle Erkenntnisse umgangen.

Andererseits haben wir eine computerbasierte Struktur zur Interpretation und zum Export entwickelt, welche es ermöglicht das rekonstruierte Wissen in mathematische Modelle und unterschiedliche Visualisierungsformate zu übersetzen.

Dadurch sind wir in der Lage, erstens Signal-Transduktionsnetzwerke detailliert zu rekonstruieren, zweitens diese Netzwerke in ausführbare Boolesche Modelle zur Verbesserung, Evaluation und Validierung dieser Netzwerke zu übersetzen und drittens diese Netzwerke als Regelbasierte Modelle zu exportieren. Daher ermöglicht rxncon die Rekonstruktion, Validierung und Simulation von umfangreichen Signal-Transduktionsnetzwerken und verbindet dadurch den Rekonstruktionsprozess mit klassischen mathematischen Modellierungsansätzen.

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Sebastian Thieme

Contribution and Collaboration

The work that I present in this thesis was supported by the Federal Ministry of Education and Research. I was supervised by Marcus Krantz and worked together with my colleagues Jesper Romers, Ulrike Münzner and Mathias Wajnberg on the rxncon project.

All chapters within my thesis are based on results of this project. I took part in the development of rxncon, the bipartite Boolean model and rule-based model. Therefore, I developed concepts for the implementation of the models within the rxncon framework and tested those concepts in collaboration with Ulrike Münzner. In addition, I contributed to the development of the rxncon language.

The project where I worked on my own were the species-reaction graph and the new implementation of the reaction graph. The regulatory graph was developed with equal contribution by Mathias Wajnberg and me. In addition, the BNGL translation and bipartite Boolean model were developed with equal contribution by Jesper Romers and me.

Sebastian Thieme

Danksagung

Zuallererst möchte ich mich bei Markus Krantz bedanken. Er gab mir die Möglichkeit in einer jungen Gruppe zu arbeiten und an den Herausforderungen, die sich mir gestellt haben zu wachsen und mich zu entwickeln. Ohne Ihn wäre diese Arbeit nicht möglich gewesen.

Ich möchte mich auch bei Ulrike Münzner bedanken. Ich bin mit Ihr durch so manche Schwierigkeit im Laufe des PhDs gegangen und habe eine Freundin in Ihr gefunden, die mir stets mit Rat zur Seite stand.

Außerdem möchte ich mich bei Jesper Romers für die tollen Diskussionen und Anregungen bedanken. Er hat mich während unserer gemeinsamen Zeit stets unterstützt, inspiriert und mich angehalten Dinge kritisch zu sehen um neue Wege zu finden.

Ein besonderer Dank geht an Katja Tummler, Friedemann Uschner und Wolfgang Giese, die mich während der sehr intensiven Schreibphase unterstützt haben und mir, neben Jesper Romers und Marcus Krantz, Anregungen zur Verbesserung dieser Arbeit geben haben.

Zudem möchte ich mich auch bei Ivo Mainz bedanken. Er hat sich immer für mich Zeit genommen, wenn ich mal wieder Probleme mit meinem Rechner hatte und mir dadurch viel nerven gespart.

Ich möchte mich auch bei Edda Klipp und der gesamten theoretischen Biophysik Gruppe für die tolle Zeit und die wundervolle Arbeitsatmosphäre bedanken. Ich habe viel durch die Zusammenarbeit mit ihnen gelernt und die Arbeit in der Gruppe immer sehr genossen.

Zum Schluss, möchte ich mich auch bei meiner Freundin Lisa Barros de Andrade e Sousa bedanken. Sie war stets an meiner Seite und hat mich wieder aufgebaut, wenn ich an mir gezweifelt habe. Ich verdanke ihr viel und werde ihr das nie vergessen.

In gedenken an meine Schwester.

Contents

CHAPTER 1

Introduction

The main goal of Systems Biology is to understand the biological mechanism behind systems like cells, tissues or organisms. The term Systems Biology dates back to 1968, when Mesarovic [1, 2] introduced Systems Biology as the linkage between biology and the general study of interrelations and interdependencies within entities (systems theory) [3] to understand biological processes. The increase in computational power and capacities, at the end of the 20th and beginning of the 21st century, made it possible to solve more complex biological questions, using computer based methods. The term computational Systems Biology was introduced by Kitano [4] and Ideker [5], defining the primary goal of this discipline as the investigation of biological systems using computational methods. More precisely, analysing biological systems by considering all interrelated and interdependent parts of a biological system – a biological system can only be understood on a system level [4, 5] (further reading [6, 7]).

1.1 Systems biology

Understanding biological systems at the system level cannot be accomplished by computational methods alone. Instead, we need a combination of experimental and computational approaches, because mathematical models simulated on computers are tools which attempt to predict the dynamics of systems but the results and the underlying assumptions have to be validated experimentally [4, 8].

We can learn a lot about biological processes by comparing *in silico* to *in vivo* experiments. Inconsistencies between simulations and experimental observations are indicative of an incomplete model reconstruction, e.g. missing regulatory knowledge or incorrect assumptions on the system. Those inconsistencies can be analysed by target experiments [9]. If we can detect the cause of discrepancies between the model and experimental results, we can systematically use the discoveries to fill those knowledge gaps. Hence, an inconsistent model can be as insightful as a consistent model, because it provides an opportunity for biological discoveries [10]. Consistent models can be used to make predictions which can be validated by experiments

and enable the exploration of new hypothesis. Therefore, both inconsistent (not showing the expected behaviour) as well as consistent (showing the expected behaviour) models can lead to model-guided experimental discoveries. The resulting discoveries can be incorporated into the previous model, increasing the information content available for the modelling tasks. The iterative improvement of mathematical models allows for better predictions of biological processes, leading to discovery of new biological mechanisms. [9, 11, 12, 13, 14, 15, 16, 17, 18]

1.2 A new era in Systems biology

Simulation-based methods received little attention in the field of biology during the early stages of computational Systems Biology, because mainly experimental methods enabled the discovery of biologically important components and mechanisms, e.g. DNA sequences and protein properties, leading to biological research with a strong focus on experimental methods [7].

During this time it was estimated that the manual annotation of a genomic sequence will require a year per person per mega base [19]. The development of automated sequence assembly and analysis methods changed everything [20, 21, 22]. In 1995, scientists were able to annotate the genome sequence of *Haemophilus influenzae Rd* and therefore to sequence the first complete genome of an organism [23]. Further development of methods for genome sequence analysis [24, 25] increased the need for experimental methods to validate predicted knowledge [26, 27, 28, 29, 30]. This development enabled scientists to decode the complete human genome sequence in 2001 [31], changing the field of life science completely. In molecular biology, new techniques were developed to produce high-throughput quantitative data, rapidly increasing the amount and complexity (in terms of information content) of the available data [32, 33, 34, 35, 36, 37, 38].

The increased quantity of omics and phenotypic data led to an increased need for dataset integration. The integration of different data types is a great challenge, which is aimed by projects as 'Big data to knowledge' (BD2k) [39]. The high amount of data we have nowadays allows us a genome-scale point of view, by linking different data sets representing the knowledge we have about biological processes [11, 13, 40]. However, we need the support of computational science to gain new biological insights by analysing, combining and interpreting the available data in a genome-scale context (further reading [41]). The newest developments and improvements in software [42, 43, 44, 45] and computational power (high performance computing environments) enable the creation and analysis of consistent models, providing useful biological insights and predictions [45, 46, 47, 48, 49].

Taken together, the understanding of biological systems can only be achieved by combining computational techniques and experimental technologies to produce more and more precise data [28, 50, 51]) and predictions [52, 53]. Only then we have the possibility to understand biological systems on their system-level by incorporating molecular biological information [4].

1.3 Yeast as model organism

Saccharomyces cerevisiae (also referred to as yeast) is a simple eukaryotic organism. In yeast, we can distinguish between different cell types with different genetic properties. The first cell type are haploid cells (one copy of each chromosome), which comprise MATa and MATalpha cells.

Those cells can respond to pheromone, enabling them to mate. In addition, a haploid cell can switch its mating type to find a suitable mating partner. However, haploid cells are not able to undergo meiosis [54]. The second cell type are diploid cells (two copies of each chromosome). Those cells cannot respond to pheromone and therefore, are not able to mate but in contrast to haploid cells they can build an ascus for sporulation, containing four haploid spores, the third cell type. [54, 55, 56]. The particular structure of spores enables yeast to survive unfavourable conditions. This cell type has also other advantages: it can be used to clean the genome from accumulated mutation; it enables the selection of the most promising gene combinations, having any advantage under certain environmental conditions; it enables mating with other haploid yeast cells of different populations, which increases genetic variety and therefore, the chance for new gene combinations [56].

In addition, we can distinguish between mother and daughter cells. Mother cells can switch mating type to mate with its daughter, leading to a clone of the cells (homozygous for all genes), which are able to conserve and reproduce beneficial genetic information (further reading [54, 57]). Mating type switches must be prevented to do genetic experiments and to grow haploid cells in the laboratory. Hence, all laboratory strains are HO mutants and cannot switch mating type (HO is a site-specific endonuclease which is required for gene conversion at the MAT locus) [54, 58].

Many essential cellular processes are similar between yeast and human, which is one of the main reasons why yeast became one of the most important model organisms to study basic molecular processes [59, 60]. Other reasons are: 1) yeast cells are easy to manipulate and they can live in many different environmental conditions, 2) yeast cells have a nucleus containing Deoxyribonucleic acid (DNA), packed into chromosomes [61], 3) yeast cells have a controlled cell division and it was shown that many genes involved in the cell division regulation in yeast have homologues in the cell division control of higher eukaryotes including human cells [62, 63]. Hence, yeast is one of the best studied model organisms and has still a great impact in gaining knowledge about biological process in eukaryotes [60, 64, 55, 65] (further reading [63]).

1.4 Signal transduction pathways

Signal transduction pathways transmit a cellular response through the cell based on extracellular signals. The cellular response to this signal leads to the activation a signalling cascade within the cell.

Insulin signal response pathway

A common example is the insulin/insulin-like growth factor signal response pathway (Figure 1.1, adapted from [66]; Supplementary Table S2), reviewed in [66], which is conserved between diverse species [67]. The main role of insulin is the regulation of the metabolism of glucose and lipids [66]. Insulin activates the receptor tyrosine kinase (IR) and the closely related type 1 insulin-like growth factor receptor (IGFR) [68]. The IR receptor in turn phosphorylates and recruit different substrates, e.g. insulin receptor substrates (IRS) and SHC-transforming protein 2 (Shc) [69, 70]. Tyrosine phosphorylated IRS binds to a number of signalling partners, e.g. phosphoinositide 3-kinase (PI3K), which has a major role in the insulin response by activating the Akt/PKB cascades. Activated Akt is important for metabolic processes, e.g. synthesis of lipids or glycogen as well as for cell survival, growth and proliferation. The

insulin receptor also phosphorylates and recruits Shc which builds a complex with the growth factor receptor-binding protein 2 (Grb2) and son of sevenless (SOS), leading to the activation of the Ras/mitogen-activated protein kinase (MAPK) pathway. The Ras/MAPK pathway also mediates metabolic process but its main function is the control of mechanisms influencing proliferation and differentiation of the cell [71].

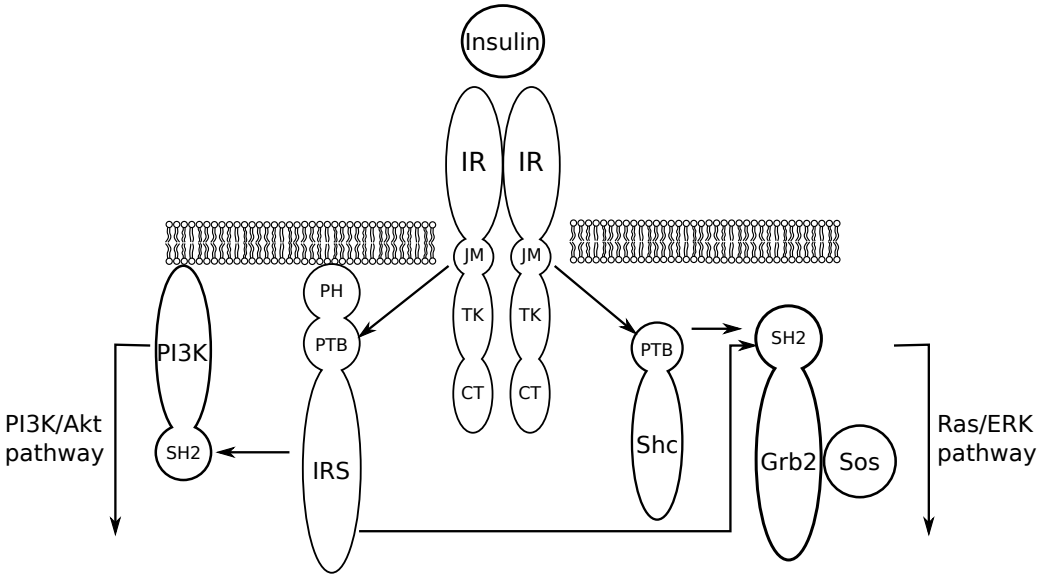


Figure 1.1: **Graphical representation of the insulin signal response pathway.** (Adapted from [71])

MAP kinases build a functional module within such a pathway that couples upstream input signals to different output responses (Figure 1.2A, adapted from [72]) [73]. It is described by a sequential activation of three kinases. The key player in this module is the 'main MAPK' itself (also referred to as extracellular signal-regulated kinases (ERK) within human cells), which regulates downstream functions. The 'main MAPK' is activated by an upstream MAPK kinase (MEK: abbreviation for MAPK/ERK kinase) that is activated by another MAPK kinase kinase (MEKK: abbreviation for MAPK/ERK kinase kinase), which in turn is activated by an input signal, e.g. ligand binding to a receptor. This pathway is conserved within eukaryotes and important for the regulation of different cellular processes, e.g. cell proliferation, mating and stress response [73].

Yeast has five distinct MAP kinases: Hog1, important during osmotic stress; Fus3, main MAPK during sex communication; Kss1, important for pseudohyphal development; Slt2 (Mpk1), important for cell-wall integrity and Smk1, important for sporulation [72, 74, 75, 76, 77, 78, 79, 80].

Pheromone response signal pathway

The pheromone response signal pathway in yeast is crucial for its sexual communication and therefore, plays a key role in the process of mating [81]. The pathway itself consists of a specific pheromone receptor Ste2/3 (Figure 1.2B; Supplementary Table S3) that binds the pheromones a- or α-factor and controls the expression of genes which are important for mating. The activation of the signal transduction pathway causes cell cycle arrest in the G1 phase, preparing the cell to from a diploid cell with a haploid cell of opposite mating type. Similar to many human hormone receptors the receptor Ste2/3 belongs to a class of seven transmembrane G-protein coupled receptors.

The binding of pheromone to the G-protein-couple receptor (Ste2/3) leads to a release of its α-subunit from the β-(Ste4) and γ-subunits (Ste18) [81, 82]. Ste4 and Ste18 (the Gβ/γ-subunit) activates the signalling branch by binding to a Ste5-Ste11 complex and to the Ste20 kinase. This allows Ste20 to bind to Cdc42 (the activated form is located at the plasmamembran) which activates the kinase activity of Ste20 and enables its auto-phosphorylation [83, 84, 85]. The recruitment of the scaffold protein Ste5 initiates a phosphorylation cascade starting with Ste11 phosphorylated by Ste20, Ste7 phosphorylated by Ste11 and ending in the phosphorylation of Fus3, the 'main MAPK' of this pathway [86, 87, 88, 89, 90]. After phosphorylation by Ste7, Fus3 is activated and translocated to the nucleus where it is able to phosphorylates and activates Far1, which is important for the cell cycle arrest in G1 phase as well as for polarized grow (Msn5p mediated export of the Far1-Cdc24 complex, targeting Cdc24 to polarity sites) [91, 92, 93, 94, 95, 96, 97]. In addition, Fus3 activates the transcription factor Ste12 (targeting over 200 genes) through phosphorylation of Dig1 and Dig2, repressing Ste12 [98, 99, 100, 101].

In summary, the binding of pheromone stimulates a signal cascade, a so-called Mitogen-activated protein kinase pathway, responsible for cell cycle arrest in G1 phase, the main transcriptional response for the mating process [77] and a reorientation of the cell towards a mating partner (towards the gradient of the pheromone) [102].

High-osmolarity-glycerol pathway

Another fundamental ability of yeast cells is the adaptation to osmotic-stress. The pathways involved in this process are crucial for the survival of yeast cells and therefore, evolutionary conserved among different species. In yeast, osmo-adaptation is regulated by the high-osmolarity-glycerol (Hog) pathway, a MAP kinase cascade that is rapidly activated under high osmolarity conditions, triggering a transcriptional response [76, 103] (Figure 1.2C; Supplementary Table S4).

The Hog pathway is controlled by two different but redundant branches, Sln1 and Sho1. Both branches are converging into Pbs2, the MAPKK of Hog1, the 'main MAPK' (reviewed in [76]) [104, 105]. The Sln1 branch consists of a osmosensor Sln,1 localised at the plasma membrane and forms, together with Ypd1 and Ssk1, a phosphorylation system [105, 106]. Active Sln1 performs auto-phosphorylation under normal conditions, whereas under osmotic stress the receptor is inactivated. The phosphoryl group of Sln1 is transferred to a receiver domain on Sln1, further to Ypd1 and finally to the receiver domain in Ssk1. Phosphorylated Ssk1 is inactive and is not able to activate the downstream MAP kinase cascade.

Under osmotic stress, unphosphorylated Ssk1 accumulates and binds to the regulatory domain of the two MAPKKKs, Ssk2 and Ssk22. The binding activates the kinase domain of the MAPKKKs and enables auto-phosphorylation, resulting in active Ssk2 and Ssk22. The active

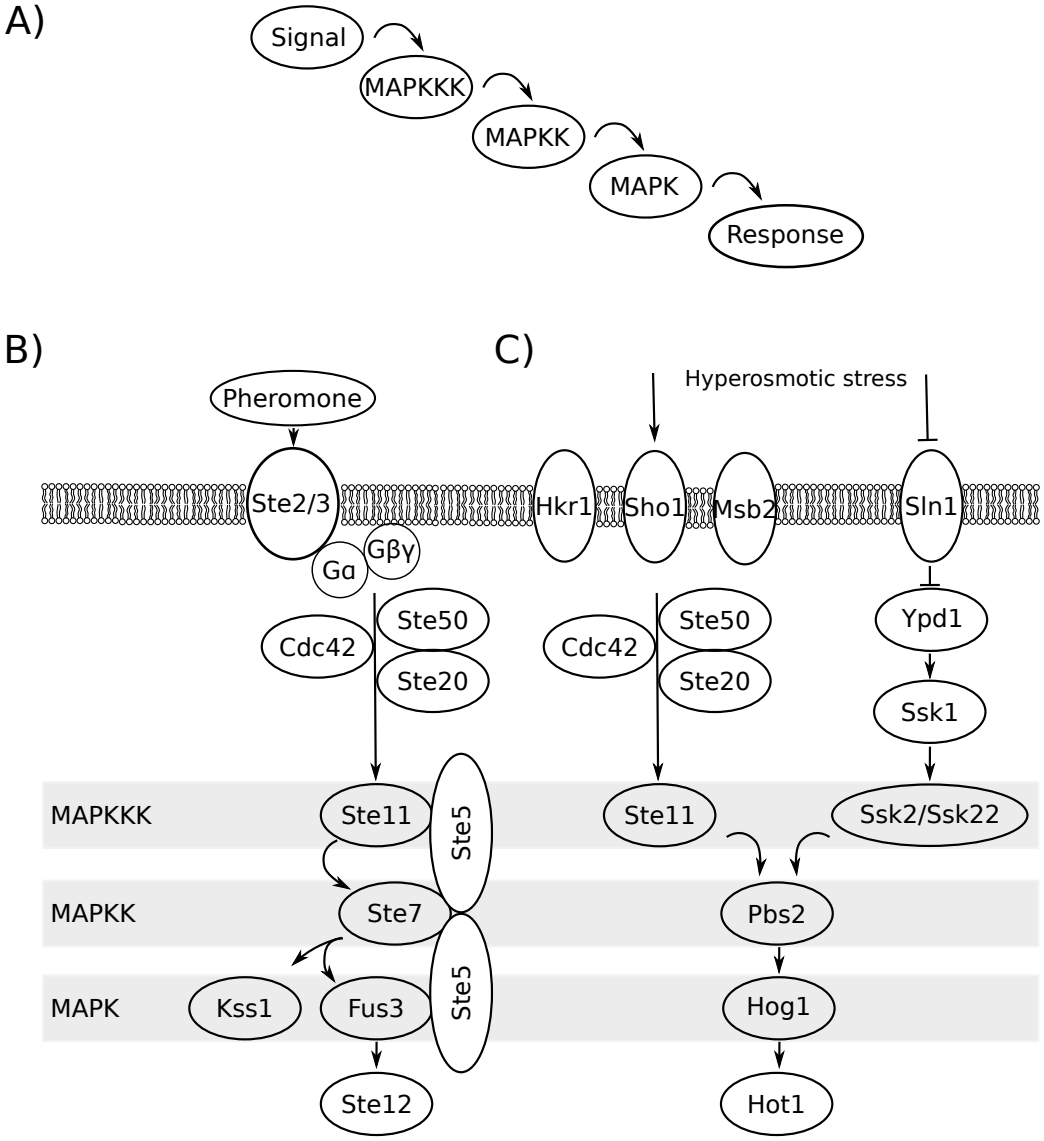


Figure 1.2: **Mitogen-activated protein kinase pathway.** A) Schematic representation of a general Mitogen-activated protein kinase pathway. B) Graphical representation of the High-osmolarity-glycerol MAPK pathway in yeast. C) Graphical representation of the pheromone response signal pathway in yeast. Figure B) and C) are adapted from [72]

Ssk2 and Ssk22 activates Pbs2 via phosphorylation, leading to the activation of Hog1 through the phosphorylation of its two residue sites Thr174 and Thy176 by Pbs2 [104, 105, 107]. The second branch is the Sho1 branch, which is controlled by two transmembrane sensors Msb2 and Hkr1 [108]. Both are mucin-like transmembrane sensors, connecting the interior with the

extracellular matrix and monitor movements between cell wall and plasma membrane (sensing the change of available water).

Sho1 is an additional transmembrane protein, localised in the membrane and acts as a scaffold protein. Sho1 recruits Pbs2 to the plasma membrane, which serves as scaffold protein for the MAPK cascade [109, 110, 111]. Through the re-localisation of Ste20 to the plasma membrane it can bind to the small Rho GTPase Cdc42 [112], which in turn brings Ste11 in close proximity to Ste20 and enables Ste20 to phosphorylate and thereby to activate Ste11. The MEKK Ste11 can now activate Pbs2, which activates Hog1 [103]. Active Hog1 is imported into the nucleus [113], where it influences gene expression by interacting and activating Hot1 [76].

The most important role of the Hog pathway is the control of glycerol accumulation during osmotic stress, acting as osmolyte. Glycerol is a by-product of yeast fermentation and produced out of two reasons: 1) osmoregulation and 2) redox-balancing [76]. The Hog pathway controls the glycerol accumulation during osmotic stress at different levels and with different time delay: 1) regulation of genes important for glycerol production, e.g. Gpd1 or Gpp1, 2) expression of Stl1 a glycerol proton symporter, allowing active glycerol uptake from the environment and accumulation in the cell interior [114, 115, 116], 3) increases of the glycerol production rate by regulating the responsible enzymes [117] and 4) regulates the activity of the glycerol exporter channel Fps1 [118].

In summary, the reduced availability of free water stimulates a Mitogen-activated protein kinase pathway with two different signalling cascades activating the same central MAPK (Hog1), which is responsible for fast accumulation of Glycerol as osmolyte, the main transcriptional response for short-term and long-term adaptation to new environmental conditions (further reading [119]).

1.5 Computational modelling

The simulation of computational models, describing signal transduction networks, can help to gain knowledge about those networks and model-guided experiments can validate biological hypothesis as well as discover new biological insights. However, before we can start simulating or analysing computational models, we have to formulate our knowledge in a computer and (ideally) human readable way. Therefore, we need approaches which can be used for models dealing with modifications and interactions within signal transduction networks, e.g. Pheromone response pathway, Hog or Insulin pathway. Those approaches should allow the user to define signal transduction networks via an adequate (standardised) reconstruction process, avoiding the combinatorial complexity problem. The reconstruction should be computer and human readable, e.g. text-based format, and the model should represent the underlying empirical data in an accurate way. Furthermore, it should be possible to simulate the underlying dynamics of the reconstructed signal transduction network [120, 121]).

In this work, I present computational approaches that can be used within a workflow to reconstruct large signal-transduction networks and to create of a quantitative models [122]. The workflow is inspired by [123] and can be divided into five steps, which I describe in the following three Chapters. In the first two steps the scope of the network has to be defined and a first seeded version is created, which can be refined by adding mechanistic information. How to build up such a mechanistic detailed qualitative system, using the new version of the rxncon language, is the topic of Chapter 2. In step three and four, this rxncon system can be translated into a ready-to-simulate bipartite Boolean model, an approach that allows parameter

free structural validation and simulation of large-scale signal transduction networks. In Chapter 3, I introduce the bipartite Boolean model and its generation, using the rxncon system. If the bipartite Boolean model behaves as expected, showing the expected outcome to a certain input signal, we can translate the reconstructed network into a quantitative model in step five, using a rule-based model, the topic of Chapter 4. In the last Chapter of my thesis I summarise and discuss my work and will give an overview on potential follow up projects.

CHAPTER 2

Network reconstruction using rxncon

In this chapter, I introduce the rxncon language 2.0 [124] an improvement of the previous developed rxncon language [121]. Based on the new syntax I show how to use the rxncon language to reconstruct biological signal-transduction networks and exemplify the main principles, e.g. elemental reactions and contingencies, on biological examples. Furthermore, I show how we handle non-elemental statements and what satisfiability of contingencies means. I demonstrate the application of the rxncon language on a simplified version of the Hog pathway and show that this approach is scalable using an already existing model of the pheromone pathway [121], which was adapted according to the changes in the language.

The first step towards the construction of a mathematical model is the reconstruction of a network, a process to formalise the available biological knowledge. The formulation of biological knowledge should be in a readable format, ideally computer and human readable [123], and should allow the exchange of models and data [125]. To enable computational analysis and to improve reusability of networks the community developed different databases and standards which can be used for network reconstruction and annotation [126, 127, 128, 129, 130, 131].

We have to translate and interpret a reconstructed network into an accurate mathematical format representing biological properties as exact as possible. Those models are then used to compute cellular states like adaptation to stress, growing conditions or system changes through cell-cell interaction. A common notion of complex systems is that a large number of simple and identical elements are interacting to produce a 'complex behaviour'. However, in signal-transduction networks we have a large number of functionally diverse sets of elements (proteins), each with a number of internal states, e.g. different modifications, interact in a selective and non-linear way to rather produce a biological meaningful than complex behaviour [4]. The combination of those sets of proteins defines a specific state or microstate within the specific state space, describing the space of all possible microstates of the system. We have to consider all possible combinations due to the lack of knowledge regarding the importance of specific combinations within a signalling process, to be accurate in our reconstruction. This leads to a large number of microstates even for small signal-transduction networks. For instance, if we consider a protein with 10 covalent modification sites (phosphorylation sites) and have no

further knowledge about this state, we have to consider a specific state space of 2^{10} different microstates. This might not be a problem if we have only one or two of these proteins in the network but often there are more as for example in the pheromone pathway. Usually we have only experimental measurements for single modifications or interactions at a time. Hence, we have the problem that the combination of internal protein states results in a much higher resolution than the empirical data. This leads to a discrepancy between the information we get from experiments and the specific states of the model we are trying to describe the data with [132]. This gap between reconstructed state space and underlying data as well as the combinatorial problem increase with an increase in network size. The resulting highly complex systems cannot be simulated anymore with reasonable computational costs, leading to arbitrary reductions and therefore, to a decrease in accuracy of the reconstructed network and the resulting model.

Established methods for reconstructing and analysing networks exist for metabolic networks, which can even handle large-scale networks [133]. Methods applied on metabolic networks can also be applied to other cellular process that could be reconstructed in biochemical detail, e.g. signal-transduction networks. The difference to metabolic models compared to signal-transduction networks is that we rather transfer information than mass. In metabolic networks the analysis occurs on a more topological level, assuming that the presence of a substrate is sufficient for a reaction to happen. In signal-transduction networks the reactions depend on the internal state of the component, meaning that a component can have multiple internal states and each state can be important for another reaction. To overcome this problem a similar approach as for metabolic networks is used by introducing an own node for each specific state, which allows the application of methods developed for metabolic networks on signal-transduction networks. However, using metabolic reconstruction methods for signal-transduction networks is limited in scalability as soon as the network size increases [132].

An alternative approach is the reaction-contingency based format. This format enables a systematised and condensed reconstruction of signal-transduction networks, by defining the network as decontextualized reactions and adding contextual constraints - contingencies - on these reactions. Similar to the rule-based approach [134, 135] (discussed in Chapter 4) only the information which is important for the reaction is considered and the remaining information is ignored - known as 'don't write, don't care' principle. This description corresponds closely to empirical data and therefore, to macroscopic states. This in turn largely avoids the combinatorial complexity during the reconstruction process as well as the discrepancies between the empirical data and the reconstruction. Hence, the reaction-contingency formalism scales well with an increase in network size and has the potential for reconstructing large-scale signal transduction networks.

Therefore, we previously developed the rxncon (reaction-contingency) language [121, 124]. The rxncon language is based on context free grammar and a strict separation of the required mechanistic building blocks: decontextualized elemental reactions, describing reactions in terms of changing elemental states, and their necessary molecular context - contingencies - expressed in terms of elemental states or combination of elemental states. The key features of the language are that: 1) rxncon states are macroscopic states, meaning that only relevant information is reconstructed; 2) the language corresponds closely to empirical data and therefore, largely avoids the combinatorial complexity problem during the reconstruction. It was previously shown that this formalism scales well with an increase in network size and therefore, is suitable to capture signal-transduction networks with full mechanistic detail [136]. However, we enhanced the rxncon language and developed rxncon 2.0, the second generation rxncon

language, improving the expressiveness and precision of the previously developed rxncon language.

2.1 The rxncon system

The rxncon system contains the knowledge about the reconstructed mechanistic processes of a signal-transduction pathway described by the rxncon language. The rxncon language is based on a collection of statements describing biochemical reactions (elemental reactions) and their contextual constraints (contingencies). The information is stored in two lists: a reaction list and a contingency list. The reaction list contains all rxncon reactions which are important for the reconstruction process and, hence, describes the reaction layer. The contingency list contains all the regulatory information of the system and describes therefore, the regulatory layer. Each individual statement is independent from each other, which allows a systematised and condensed reconstruction of signal-transduction networks. Each statement within the rxncon language should be based on experimental facts, which can be taken from, e.g. literature. The semantics of the rxncon language makes use of different concepts [124]: specification, elemental states, elemental reactions and contingencies. To ensure a good understanding I formalise the rxncon language and describe its key concepts. Note, the different classes used in our implementation (see appendix section Python library) are images of these concepts.

2.1.1 Specification

A molecule specification is a central building block of the rxncon language. It is an element for both concepts, rxncon reactions and rxncon states. A specification defines the type and the structure of a molecule and consists of up to three parts: a component, a structural index and a locus.

The component denotes the type of the specification, e.g. a protein, gene or mRNA as well as the name. The name is defined as a sequence of alphanumeric characters, starting with a letter but not ending with *Gene* or *mRNA*, except this is intended, e.g. to define molecules of a transcription or translation reaction. If a component name ends with mRNA or gene, it will automatically be assumed that the component belongs to the respective class. This relation ensures a precise interpretation of rxncon reactions relying on certain concepts, e.g. translation and transcription.

The locus defines a location on the molecule, e.g. a domain or a residue. This in turn enables a precise definition of which specific functional parts of the molecule interact with each other and therefore, allows a detailed reconstruction of the mechanistic processes within the reconstructed network. Each locus has a defined resolution, depending on the locus information: 1)'component level', a component without any locus information, 2)'domain level', a component with domain locus information and 3)'residue level', a component with residue locus information. Each specification has to be defined at least on the 'component level', which allows an accurate description of experimental knowledge. A combination of the different levels is possible, however, the resolution of the specification is defined as the most precise one, meaning that as soon as a residue is specified the resolution of the specification is 'at the residue level'.

Molecules can be defined on different levels of resolution, e.g. domain level or residue level. To be able to find overlapping specifications we defined a set relation between different

specifications - superset and subset relation - where a superset is defined as an outer set containing different subsets. A specification `A` is a subset of specification `B` if: 1) the components of `A`, `B` and the structure indices are identical, 2) the resolution of `A` is equal or higher than the resolution of `B` and 3) the locus information of `A` and `B` matches given that both locus information are available. For instance, `A_[x(r)]` is a sub-specification of `A_[x]`, because the components are identical, the locus information matches since both have a domain called `x` and the resolution of A_[x(r)] is higher than the resolution of`A_[x]`. Furthermore, specification `A` is a superset of specification `B` if and only if `B` is a subset of `A`. The subset and superset relation is useful for later handling of specifications, e.g. to expand non-elemental states or to calculate the complement of an elemental state (discussed later).

2.1.2 Reaction terms

Reaction term (or rxncon reaction) denotes which property of a molecule (modified residue, bound domain) changes, without resorting the microstate description. The syntax of a reaction term contains three parts: a Subject, a Predicate and an Object (Figure 2.1A). A rxncon reaction consists of two specifications (reactants) and one reaction type, meaning that the first specification (Subject) acts (Predicate) on the second specification (Object). An exception are output reactions that describe global quantities, e.g. the turgor pressure (Figure 2.1B). Depending on the reaction type, the meaning of Subject and Object can differ, e.g. for interaction reactions Subject and Object are interacting with each other while for synthesis reactions the Object will be synthesised by the Subject (Table 2.1). The elemental resolution of a reaction term

Table 2.1: Reaction types. rxncon reactions differ in their elemental resolution and in the number and kind of elemental states they process.

Reaction type	Description	Elemental resolution
Unidirectional covalent modification	A single molecule is modified	Subject: component level; Object: residue level
Bidirectional covalent modification	Two molecules are modified	Subject: residue level; Object: residue level
Interaction	Two molecules bind to each other via a bond	Subject: domain level; Object: domain level
Synthesis	Creates all neural rxncon states of a certain component	Subject: component level; Object: component level
Degradation	Destroys all elemental states sharing the same component, except if the reaction is regulated	Subject: component level; Object: component level
Output reaction	N/A	N/A

depends on the reaction type. A reaction term is defined on an elemental resolution if each

specification is defined on the required level of resolution (Table 2.1). We recommend to use only elemental reactions to be as precise as possible during the reconstruction process.

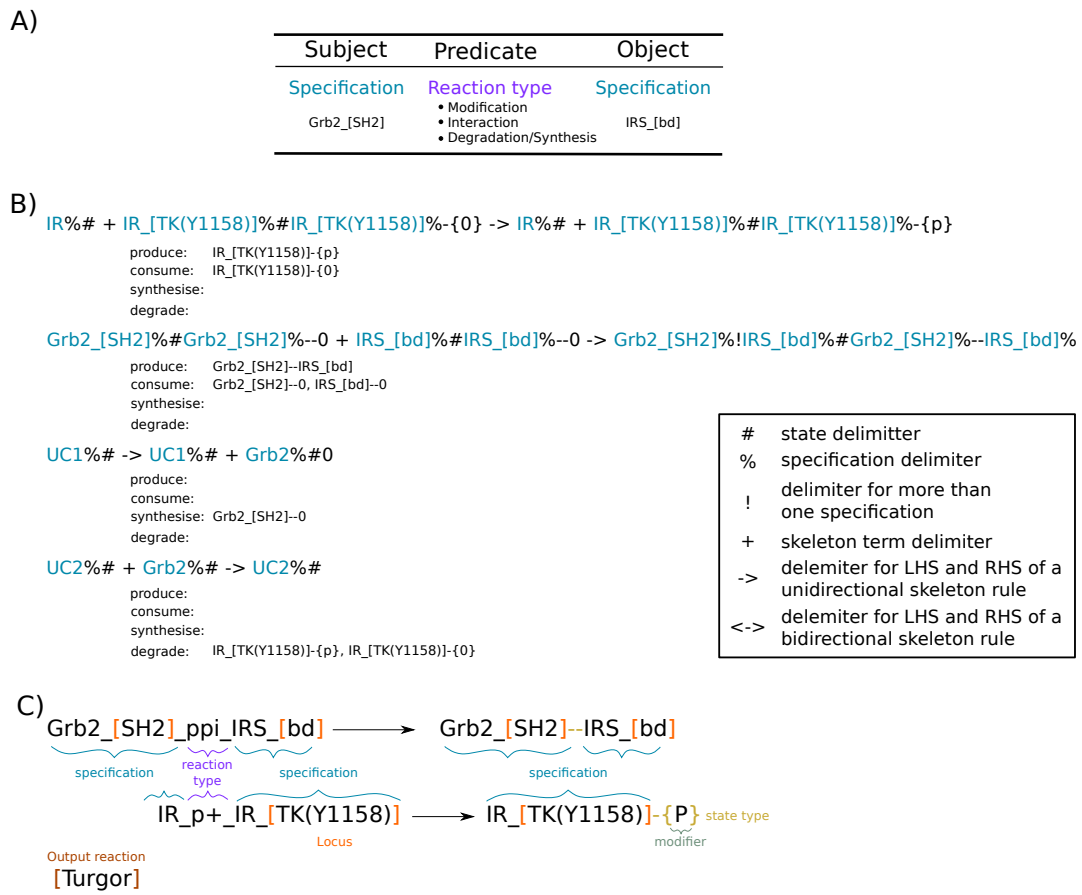


Figure 2.1: **The reaction term.** A) Overview on the syntax of a rxncon term. B) Example of four different skeleton rules, defining four different elemental reactions. From the top to bottom: the first skeleton rule defines a modification reaction, the second an interaction reaction, the third a synthesis reaction and the fourth a degradation reaction. C) Example of a protein-protein interaction reaction, producing an elemental interaction state, a phosphorylation reaction, producing a phosphorylated elemental state and an output reaction.

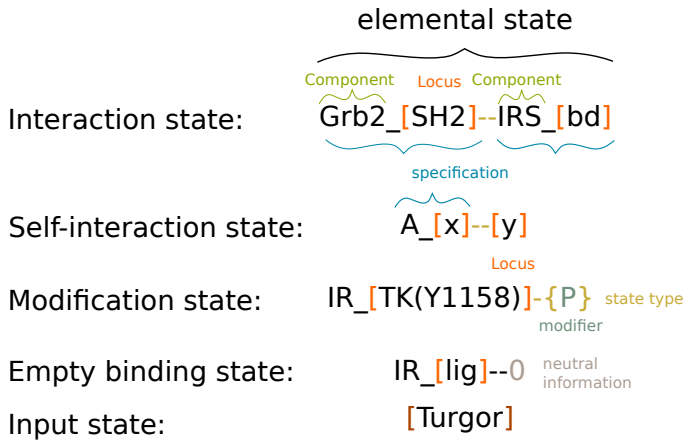
Each reaction term has its own semantic, which is given by the skeleton rule. The skeleton rule describes the general structure of the reaction term. A skeleton rule consists of different skeleton terms, e.g. skeleton terms on the left-hand site (LHS) that are transformed into skeleton terms on the right-hand side (RHS) of the rule. Those skeleton terms are composed of: zero or more components, representing molecules and zero or more elemental states, defining the internal states of the components. The skeleton rule can be used to retrieve additional properties of a reaction term: the production, consumption, synthesis and degradation of rxncon states (Figure 2.1B). Note, that an elemental reaction only produces, consumes, synthesises or degrades rxncon states (Figure 2.1C), except output reactions that do not have a skeleton rule

and therefore, do not produce, consume, synthesise or degrade any state term.

2.1.3 rxncon states

A rxncon state is defined as an independent observable quantity, e.g. phosphorylated protein or protein complex. It either consists of two specifications (e.g. protein complex) or one specification and a rxncon state property (e.g. phosphorylated protein) or one specification and one locus (e.g. two domains of the same protein interacting with each other). We separate rxncon states into three categories: 1) one molecule is involved, e.g. modification states, self-interaction states and empty-binding states, 2) a pair of molecules is involved, e.g. interaction states or 3) no molecule is involved, e.g. input states (Figure 2.2A). The resolution of a rxncon state is given by the resolution of its specifications. A special case is the self-interaction state where the second part is not a specification but a locus. The resolution of a specification (or a locus in the case of self-interactions) is elemental if the locus information is uniquely defined on the level of the elemental resolution given by the class of the rxncon state. (Figure 2.2A). If every specification (or locus) is at an elemental resolution, the rxncon state is referred to as an elemental state. Furthermore, every rxncon state has a neutral rxncon state as counterpart (Figure 2.2B). Note, rxncon states containing more than one specification are equivalent if one of their permutations of specifications are equal, e.g. `A_[b]-B_[a]` is equivalent to `B_[a]-A_[b]`.

A)



B)

Elemental state	Neutral counterpart
Grb2_[SH2]--IRS_[bd]	Grb2_[SH2]--0
A_[x]--[y]	A_[x]--0
IR_[TK(Y1158)]-{P}	IR_[TK(Y1158)]-{0}

Figure 2.2: **The rxncon states.** A)Overview of different types of rxncon states and their composition. B) Example of neutral counterpart for non-neutral elemental states.

rxncon states can also be a superset or subset of other rxncon states. Therefore, the superset and subset definition is inherited from specifications. A rxncon state S1 is a subset of a rxncon state S2 if: 1) they belong to the same rxncon state class, 2) all non-specification properties are equal and 3) all specifications of S1 are a subset of the specifications of S2. For instance, `A_[m]-B_[n]` is a subset of `A-B`.

2.1.4 Building up the state space

The reaction layer describes all rxncon reactions within the rxncon system. If we build up the reaction layer, we extend the information about the described molecules and therefore, the possible configurations a molecule can be in, meaning that if we add rxncon reactions to our rxncon system, we build up the state space of our model. The literature can be used to identify elemental reactions, which connect components with each other. For instance, for the Hog pathway we will find statements like *'... activate Pbs2, which in turn phosphorylates (on Thr174 and Thy176) and activates Hog1 ...'* [137], which describes the phosphorylation of Hog1 at two sides (Thr174 and Thyr176) by Pbs2. Hence, if we translate this into the rxncon language, we would get two different rxncon reactions that differ only in the object residue. This statement will be defined as `Pbs2_p+_Hog1_[(Thr174)]` and `Pbs2_p+_Hog1_[(Thy176)]` using a string representation of a rxncon reaction (see appendix rxncon input formats). The Subject of this rxncon reaction is Pbs2, the Predicate is p+ (reaction type abbreviation for phosphorylation) and the Object is Hog1 with the residue Thr174 or Thy176. The elemental reaction `Pbs2_p+_Hog1_[(Thr174)]` creates a specific elemental state `Hog1_[(Thr174)]-{p}`, which has the elemental state `Hog1_[(Thr174)]-{0}` as counterpart. Hence, the rxncon reaction adds a new rxncon state property (the rxncon state is phosphorylated at a certain residue) to the system.

Furthermore, we can find the statements like *' ... Pbs2 contains a region that strongly interacts with Hog1, which we have named Hog1-binding domain 1 (HBD-1) ... the Hog1 CD domain is indeed required for interaction between Hog1 and its activator Pbs2 ...'*, which means that Hog1 and Pbs2 bind to each other at the HBD1 and CD domain, respectively [138]. This adds another rxncon reaction `Pbs2_[HBD1]_ppi+_Hog1_[CD]` to our system, further increasing the state space for Hog1: Hog1 is bound to Pbs2 (`Pbs2_[HBD1]-Hog1_[CD]`) and Hog1 is not bound to Pbs2 (`Hog1_[CD]-0`). We now have four possible configurations of Hog1: 1) Hog1 is phosphorylated and bound to Pbs2, 2) Hog1 is phosphorylated and not bound to Pbs2, 3) Hog1 is not phosphorylated and bound to Pbs2 or 4) Hog1 is not phosphorylated and not bound to Pbs2. The state space for Hog1 increases exponentially, because every new rxncon reaction acting on Hog1 adds a new property to the Hog1 molecule. However, the reconstruction only increases linear, since we add only one new line for every rxncon reaction. An exception are synthesis reactions and degradation reactions. Those rxncon reactions do not change the state space, because they do not add properties to molecules. Since the reconstruction is on a single molecule-level, we can also add rxncon reactions creating mutually exclusive rxncon states. These are rxncon states of the same type, containing the same molecules, which are defined on the same elemental resolution but differ in a property, e.g. modification or binding partner. For instance, the rxncon state `Hog1_[(Thr174)]-{p}` is mutually exclusive of its neutral form `Hog1_[(Thr174)]-{0}`.

Synthesis and degradation reactions are different from other rxncon reactions like phosphorylation rxncon reactions (Figure 2.1B)), because both are acting on a component level. Synthesis reactions simultaneously synthesise all neutral states of a certain component (so called fully-neutral state). This is a rxncon system property, because the fully-neutral state depends on

the rxncon reactions defined in the reconstruction process and their produced and consumed rxncon states. An unregulated degradation reaction degrades all rxncon states containing a certain component (regulated degradation are discussed below). The degradation of a complex will lead to the degradation of one partner of the complex, carrying the component that is target of the degradation, and the release of the partner, not carrying the specific component, into an empty-binding state. If both interacting partners have the same component (homo-dimer), means that the complete complex will be degraded.

2.1.5 Non-elemental reactions

During the reconstruction process, we want to describe the experimental knowledge as precise as possible. However, we often do not know the precise residue which is phosphorylated but only the domain which contains the residue. This is handled during the building process of the rxncon system. rxncon reactions which are not elemental will get default domains and residues according to the type of the rxncon reaction. The default name of the domain or residue is the same name as for the interaction partner (Subject, Object). For instance, defining a rxncon reaction as `Hog1_p+_Hot1` will automatically result in `Hog1_p+_Hot1_[(Hog1)]` because the phosphorylation reaction is elemental on the residue level for the Object and on the component level for the Subject. The rxncon reaction `Hog1_ppi_Hot1` will result in `Hog1_[Hot1]_ppi_Hot1_[Hog1]` [114, 139], because a protein-protein interaction is elemental if both reactants are defined on a domain level.

2.1.6 Reducing the reconstructed state space by contextual constraint

The context for a rxncon reaction event is given by contingencies (contingency term). These constraints decrease the state space by increasing the information content within the reconstruction. The general syntax defining a contingency term is Target, contingency type, Effector (Figure 2.3). We distinguish between two categories of contingency terms: reaction-contingency terms and Boolean-contingency terms.

Table 2.2: An overview of predefined contingnecies.

Contingency type name	Contingency type sign	Description
Strict	`!`	A Target requires an Effector
	`x`	A Target is inhibited by an Effector
Quantitative	`k+/k-`	Quantitative contingency describing a positive or negative change of a rate value which switches between two non-zero values
No effect	`0`	An Effector has no effect on a Target
Unknown	`?`	The effect of an Effector on a Target is unknown

The first category of contingency terms, describes contingency terms as contextual constraints on

rxncon reactions, which are direct relationships (Contingency Type) between a rxncon reaction (Target) and a rxncon state or Boolean-contingency term (Boolean statement; Effector). Those Contingency Types are divided into strict (qualitative), quantitative, no effect and unknown Contingencies (Table 2.2).

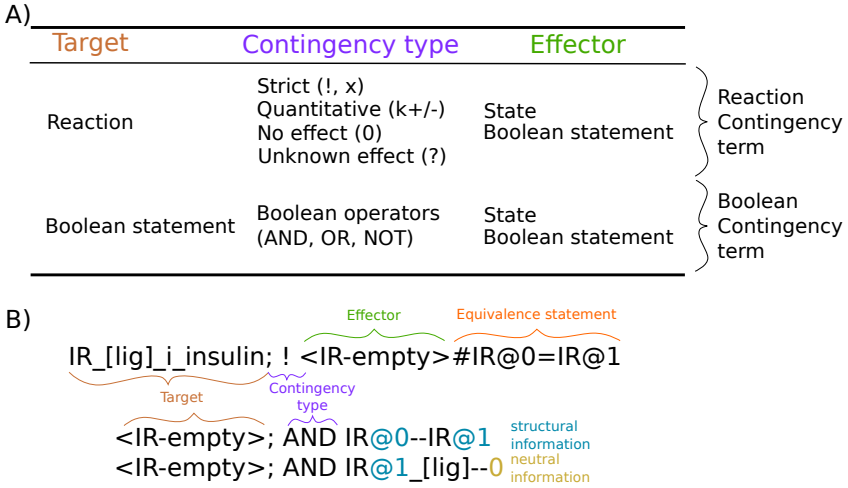


Figure 2.3: **The contingency term.** A) Overview about the different parts within a contingency term. B) Example of a reaction-contingency term with a structured rxncon complex as Effector.

The second category of contingency terms, describes the relationship between a Boolean statement (Target) and rxncon states or Boolean statements (Effectors), using the Boolean logic (Figure 2.3A). A Boolean-contingency term has a name of alphanumeric characters surrounded by pointy brackets and consists of Boolean operators (Figure 2.3A; Boolean Operators are discussed in detail in Chapter 3), which connect the different Effectors. In comparison to reaction-contingency terms, where all Effectors are linked with `AND` operators, Boolean-contingency term allow the use of all Boolean operators (`AND`, `OR` and `NOT`). Hence, Boolean-contingency terms can describe more complex relationships between Effectors, allowing to describe biological complexes.

Contingencies can be added by searching for information on how rxncon reactions are regulated. For instance, the statement *'... activate Pbs2, which in turn phosphorylates (on Thr174 and Tyr176) and activates Hog1 ...'*, identifying the phosphorylation of Hog1 by Pbs2 also states that the activation of Pbs2 is a requirement for the phosphorylation of Hog1 [137]. In addition, we can find the statement *'... phosphorylation sites required for Pbs2p activation, namely, Ser514 and Thr518 ...'*, indicates that the phosphorylation of the residues Ser514 and Thr518 are required to activate Pbs2 [140]. The reconstruction looks like:

Target	Contingency type	Effector
`Pbs2_p+_Hog1_[(Thr174)]`	!	`Pbs2_[(Ser514)]-{p}`
`Pbs2_p+_Hog1_[(Thr174)]`	!	`Pbs2_[(Thr518)]-{p}`

The Target of this contingency is the rxncon reaction Pbs2_p+_Hog1_[(Thr174)], the Contingency type is ! (absolute requirement) and the Effector are the phosphorylated states Pbs2_[(Ser514)]-{p} and Pbs2_[(Thr518)]-{p}. Note, that the Effector of a reaction contingency term has to be connected to the reactants of the Target, either directly or indirectly (discussed later).

2.1.7 Defining rxncon complexes

The rxncon language enables the definition of complexes. Complexes can be constructed by combining two or more contingency statements directly, e.g. the two statements `Pbs2_[(Ser514)]-{p}` and `Pbs2_[(Thr518)]-{p}`:

Target	Contingency type	Effector
`Pbs2_p+_Hog1_[(Thr174)]`	!	`<pbs2-active>`
`<pbs2-active>`	AND	`Pbs2_[( Thr518)]-{p}`
`<pbs2-active>`	AND	`Pbs2_[( Ser514)]-{p}`

This enables the description of arbitrary complex contextual constrains.

2.1.8 Non-elemental contingencies

Contingencies may contain non-elemental states, e.g. a modification requirement without specifying the residue, because this information is not known or irrelevant for the regulation of the Target. If such non-elemental contingencies are defined, they get expanded into disjunctive elemental states to enable further analysis. To find all candidates, which get included into this expansion, we have to consider the entire rxncon system, meaning that we have to know all the rxncon reactions of the rxncon system to expand such non-elemental states. The For example, the Effector:

Target	Contingency type	Effector
`Pbs2_p+_Hog1_[(Thr174)]`	!	`Pbs2-{p}`

is non-elemental and will be expanded into the contingency term:

Target	Contingency type	Effector
`Pbs2_p+_Hog1_[(Thr174)]`	!	`<boolean>`
`<boolean>`	OR	`Pbs2_[( Thr518)]-{p}`
`<boolean>`	OR	`Pbs2_[( Ser514)]-{p}`

Note, that this is not a biological valid statement but a valid statement in the reconstruction process.

2.1.9 Structured contingencies

A complex described by contingencies, can have multiple subunits containing the same molecule, which leads to ambiguities, because we do not know if both molecules are the same or different. Additional ambiguity is added, when combining different contingencies, e.g. nested Boolean-contingency terms (Boolean statements that contain other Boolean statements) if we use a homo-dimer or a rxncon reaction acting on a reactant with the same name. This ambiguity can be resolved by using structured complexs. Therefore, we introduce structural indices. The structural index of a Boolean statement is defined in a namespace, which is labelled by the name of the Boolean statement. Within such a namespace every structure is well defined by the indices of the specifications. To assign the specifications between Boolean statements or Boolean statements and reactants of a reaction term we have to map the indices between the namespaces of the different Boolean statements or reaction terms. If we have to consider multiple Boolean statements on the same Target, e.g. different complexes influencing the same reaction term, the namespace has to be merged to obtain unambiguous molecules. To ensure a unique mapping of specifications we add an equivalence statement to the Boolean statements (Figure 2.3B). The first element after the name of the Boolean statement refers to the namespace we are in (Target) and the second refers to the namespace of the Effector (Boolean statement), which we want to assign to the first element. Note, the namespace is a structure to organise the molecules within a contingency term, allowing the reuse of the same molecule name in different context. The notation <namespace>'.'<protein>@<structure index> can be used to refer directly to a certain component, where the '.' separates the different namespaces. Within the namespace of the rxncon reaction we define that the first reactant (Pbs2) has always the structure index 0 and the second reactant (Hog1) has always the structure index 1. In the Hog1 example, the introduced equivalence statement `<pbs2-active>Pbs2@0=Pbs2@1` affirms that Pbs2 in the rxncon reaction is the same as the Pbs2 with structural index 1 in the Boolean <pbs2-active>:

Target	Contingency type	Effector
`Pbs2_p+_Hog1_[(Thr174)]`	`!`	`<pbs2-active>Pbs2@0=Pbs2@1`
`<pbs2-active>`	`AND`	`Pbs2@1_[(Thr518)]-{p}`
`<pbs2-active>`	`AND`	`Pbs2@1_[(Ser514)]-{p}`

This is useful in larger systems to reuse contingencies, e.g. multiple phosphorylations as precondition for different rxncon reactions. For the Hog1 example we can find two critical constraints for the phosphorylation of Hog1 by Pbs2: 1) the activity of Pbs2 and 2) that Pbs2 and Hog1 are bound to each other [137, 138]:

Target	Contingency type	Effector
Pbs2_p+_Hog1_[(Thr174)]	!	<hog1-p>Pbs2@0=Pbs2@2
<hog1-p>	AND	Pbs2@2_[HBD1]-Hog1_[CD]
<hog1-p>	AND	<pbs2-active>Pbs2@2=Pbs2@1
<pbs2-active>	AND	Pbs2@1_[(Thr518)]-{p}
<pbs2-active>	AND	Pbs2@1_[(Ser514)]-{p}

The equivalence assignment <hog1-p>Pbs2@0=Pbs2@2 states that the Pbs2 in the rxncon reaction is the same as the Pbs2 with the structural index 2 in the Boolean statement <hog1-p>. In addition, the assignment <pbs2-active>Pbs2@2=Pbs2@1 states that Pbs2 with the structural index 1 in the Boolean statement <pbs2-active> is the same as the Pbs2 in the namespace of <hog1-p> with structural index 2. This leads to the equivalence: Pbs2@0 = <hog1-p>.Pbs2@2 = <hog1-p>.<pbs2-active>.Pbs2@1.

We handle all contingency Effectors on a certain reaction term internally by Boolean statement of ANDs. To ensure a unique identifiability of the different molecules we assign a default structure if no structure or no equivalence statement is given. If no structure is given, we assume that all molecules with the same name are pointing to the same molecules defined in the rxncon complex as well as in the Target. If a structure is given but no equivalence statement, we assume that the molecules with structural index differ from molecules with different or no structural index as well as from the reactants. Hence, after constructing a rxncon system every molecule within the reconstructed rxncon system has a unique index and every Effector of a contingency term a unique structure.

2.1.10 Input states and output reactions

The phosphorylation of Hot1 leads to a number of downstream reactions, e.g. the accumulation of glycerol [114, 115]. Glycerol acts as osmolyte and increases the turgor pressure of the cell, which is required for growth [114]. To describe this physiological effect or other complicated regulatory mechanism, which are not in the scope of the reconstruction we can use global output reaction and global input states as black boxes. Therefore, we can define that the phosphorylated form of Hot1 (Hot1-{p}) starts the different downstream processes in our reconstruction. Hence, we can included a global output reaction [Turgor], which requires phosphorylated Hot1, e.g. [Turgor]; ! Hot1-{p}. The increased turgor pressure is required for the auto-phosphorylation reaction of Sln1. This can be realised by including an input state [Turgor], e.g. Sln1_ap+_Sln1_[(r)]; ! [Turgor]. Input states and output reactions of the same name are related to each other, meaning that the input state [Turgor] is depends on the output reaction [Turgor].

2.2 Ensuring a valid rxncon system

A rxncon system consists of one or more rxncon reactions and zero or more contingencies. After loading the rxncon reconstruction we need two additional steps to ensure a valid rxncon system: 1) Finalising step and 2) Validating step. Within the Finalising step we expand all non-elemental contingency terms, we structure all non-structured contingency Effectors and expand the FullyNeutral state if available (applies only to synthesis reactions). The Validation step follows the Finalising step and checks the consistency of the rxncon system. A rxncon system is consistent if: 1) all elemental states appearing in a contingency Effector are produced, consumed or synthesised by an elemental reaction, 2) all elemental reactions that are Target of at least one contingency are defined within the list of rxncon reactions, 3) all Effectors within the list of regulatory constraints are valid and 4) all reaction-contingency terms have at least one satisfiable solution. Since, both steps are depending on properties of the rxncon system they can only be processed after the complete rxncon system is known.

2.2.1 Valid Effector

A Boolean-contingency term has four different types of Effectors - rxncon state and the Boolean terms `NOT`, `AND` and `OR` - and those Effectors must fulfil different constraints to be valid (Table 2.3).

Table 2.3: The validity of Effectors within a contingency term.

Effector	Handshake molecule	Validity
rxncon state	Components of the rxncon state	Always
Boolean term `NOT` on Effector	Depends on the Effector the Boolean term `NOT` is applied on	If the Effector, the Boolean term `NOT` is applied on, is valid
Boolean term `AND` on different Effectors	The union of the handshake molecules of the Effector	1) All Effectors are valid 2) The intersection of the handshake molecules of the Effectors is not empty
Boolean term `OR` on different Effectors	The intersection of the handshake molecules of the Effector	1) All Effectors are valid 2) The intersection of the handshake molecules of the Effectors is not empty

We define that a rxncon state, representing the most basic Effector, is always a valid Effector. The validity of a Boolean-contingency term depends on the validity of its Effector. If the Effector is not a rxncon state, we have to make sure that the nested Boolean terms `NOT`, `AND` and `OR` are valid Effectors. An Effector can be used to describe a rxncon complex by a Boolean term `AND`. Every molecule defined within this rxncon complex can be used to establish contact to a rxncon reaction (so called handshake molecules). To make sure that the Effector, describing an `AND` rxncon complex is valid all rxncon states within the rxncon complex have to be fully connected

to each other, e.g.:

Target	Contingency type	Effector
A_[x]_ppi_B_[y]	!	<bool>
<bool>	AND	A_[c]-C_[a]
<bool>	AND	C_[d]-D_[c]

shows that all molecules defined in the rxncon complex are connected via A with the rxncon reaction. In case of multiple rxncon complexes within an AND rxncon complex, each sub-complex has to be connected to the rxncon reaction via one of its handshake molecules, e.g.:

Target	Contingency type	Effector
A_[x]_ppi_B_[y]	!	<bool>
<bool>	AND	A_[c]-C_[a]
<bool>	AND	C_[d]-D_[c]
<bool>	AND	B_[e]-E_[b]

shows two defined sub-complexes connected to the reaction.

If a rxncon complex is defined by a Boolean term OR, there must be a shared handshake molecule over all rxncon states within the rxncon complex. If this is not the case, one handshake molecule of each rxncon state, used within the rxncon complex should be connected to the reaction term. The Boolean term NOT can only be applied on one Effector and hence, is valid if this Effector is valid. These additional constraints ensure that Boolean statements are still valid, even if they are reused in a different context within other contingency terms.

2.2.2 Satisfiability

All reconstructed contingency terms within a rxncon system have to be satisfiable. The Boolean satisfiability problem (also referred as SAT) describes the problem of determining if a solution for a certain Boolean statement exists. In case a solution does not exists, the Boolean statement will be stated as unsatisfiable, e.g. in perspective of rxncon the Effector term of:

Target	Contingency type	Effector
<bool>	AND	A_[c]-C_[a]
<bool>	AND	<notBool>
<bool>	NOT	B_[e]-E_[b]

is satisfiable because there exists a combination of True and False evaluating the Boolean statement to True (A_[c]-C_[a]= True, B_[e]-E_[b]=False), whereas:

Target	Contingency type	Effector
<bool>	AND	A_[c]-C_[a]
<bool>	AND	<notBool>
<bool>	NOT	A_[c]-C_[a]

is unsatisfiable. SAT is an NP-complete problem, meaning that there is no algorithm that efficiently solves all SAT problems. However, there exist heuristic SAT-algorithms that are able to solve many practical boolean satisfiability problems sufficiently as, e.g. picoSAT [141, 142]. This issue is further discussed in Chapter 3.

To find valid solutions for a reconstructed contingency, we have to fulfil additional constraints on the contingency terms. First, within a solution the elemental states should not be mutually exclusive. Second, the effector should be valid and third, if the Target of the respective Boolean statement is an elemental reaction the rxncon states influencing the rxncon reaction need to be connected to the reactants of this rxncon reaction. In case, a component of a specification does not map to the reactant, there needs to be at least one path from a reactant to the respective specification over a bound rxncon state. Note, not every solution of a Boolean-contingency term has to be a valid solution. It is sufficient that at least one solution is satisfiable, meaning that it contains no mutually exclusive rxncon states and is connected.

To evaluate if the reconstructed contingencies are satisfiabil, we expand all contingencies to elemental contingencies, because elemental reactions produce, consume, synthesise (in case of general synthesis) and degrade (if the degraded component has internal states) only elemental states. The expansion of non-elemental states results in either a Boolean OR of elemental states or a single elemental state. The resulting expression can then be solved. The Effector of reaction-contingency terms consist either of elemental states or Boolean statements, represented by Boolean-contingency terms (describing complexes). The verification is done by linking the implementation of rxncon to picoSAT [142], included in the python package PyEDA [143]).

2.2.3 Non-satisfiable contingencies

Contingencies can contain contradictory statements which are not satisfiable. In this context satisfiable means that we can construct at least one statement from the contingencies, which does not contain mutually exclusive rxncon states and ensures that all molecules are connected to at least one reactant. For instance, if we define the following statements:

Target	Contingency type	Effector
Hog1_p+_Hot1_[(hog1)]	!	Hog1_[(Thy176)]-{p}
Hog1_p+_Hot1_[(hog1)]	!	Hog1_[(Thy176)]-{0}

which can also be written as:

Target	Contingency type	Effector
Hog1_p+_Hot1_[(hog1)]; <nsc>		
<nsc>	AND	Hog1_[(Thy176)]-{p}
<nsc>	AND	Hog1_[(Thy176)]-{0}

We expect that the rxncon reaction happens if Hog1 is both, phosphorylated and unphosphorylated, on the same residue at the same time which is contradictory, because both are mutually exclusive. Such statements will be rejected. The same holds for non-connected contingencies:

Target	Contingency type	Effector
Hog1_p+_Hot1_[(hog1)]	!	<nsc>
<nsc>	AND	Hog1_[(Thy176)]-{p}
<nsc>	AND	Hog1_[(Thy176)]-{0}
<nsc>	AND	Pbs2@2_[(T518)]-{p}

Here Pbs@2 refers to a protein which is not connected to either of the reactants Hog1 or Hot1. If there is no overlap with the rxncon reaction, the contingency has to be rejected, because there is no relation between the different molecules.

2.3 Reconstructed models and visual validation

To demonstrate a complete reconstruction in rxncon syntax, we applied the reconstruction on a simplified version of the human insulin pathway [66] and the Hog mitogen-activated protein kinase pathway for *Saccharomyces cerevisiae* (taken from [144]). The reconstruction of the Hog pathway consists of 11 rxncon reactions and 6 contingencies (Supplementary File SF1). The reconstructed insulin pathway consists of 23 rxncon reactions, defining the reaction layer and 20 contingencies, defining the regulatory layer (Supplementary File SF2). The reaction layer of a rxncon reconstruction can be visualised as a rxncon reaction graph (Figure 2.4)

2.3.1 Reaction graph

Large-scale networks can be difficult to understand. Therefore, it is important to provide a possibility of visualising the underlying information in a condensed way, enabling a first visual inspection of the network. The reaction graph represents the topological structure of the reconstructed network (reaction layer), using directed and undirected edges of the reconstructed network. We consider three categories of nodes: graphical-component nodes, representing the

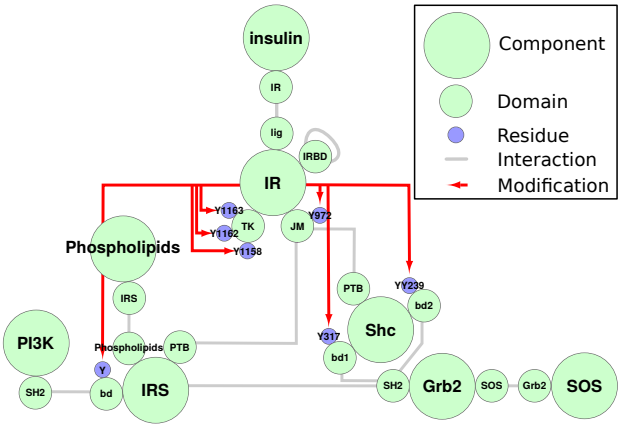


Figure 2.4: **Reaction graph of the insulin pathway.** The rxncon reaction graph visualises the reaction layer of the reconstructed network. In this graph insulin (top component) is connected to PI3K as well as to SOS (two bottom-most components).

component information of a specification as well as graphical-domain nodes and graphical-residue nodes, representing the Locus information. The inclusion of locus information makes the reaction graph similar to contact maps [135, 145]. Both graphs show only the structural information and leave out any regulatory information, compared to extended contact maps, which can consider regulatory information, e.g. if a binding depends on a modification. The edges show the relation between the different specifications with respect to the defined rxncon reactions. We consider five types of edges: undirected interaction edges, unidirectional modification edges, bidirectional modification edges as well as degradation and synthesis edges. The source and target of the edges is defined by the resolution of the rxncon reaction (Table 2.4).

Table 2.4: Reaction graph edges.

Edge type	Directionality	Description
Interaction	Undirectional	edge from a domain level to a domain level
Unidirectional modification	Unidirectional	Pointing from a component level to a residue level
Bidirectional modification	Bidirectional	Pointing from a residue level to a residue level
Synthesis	Unidirectional	Pointing from a component level to a component level
Degradation	Unidirectional	Pointing from a component level to a component level

The edges correspond to rxncon reactions and a chain of rxncon reactions is required but not sufficient for information transfer.

2.3.2 Species-Reaction graph

The species-reaction graph represents a detailed view of the regulatory mechanism of a reconstructed network and shows the information flow through the network. In contrast to the reaction graph the species-reaction graph includes the information about the causality between nodes of the network. We consider five categories of nodes: 1) graphical-reaction nodes, representing elemental reactions, 2) graphical-state nodes, representing elemental states, 3) graphical-Boolean nodes, representing Boolean contingency terms, 4) graphical-component nodes, representing global quantities, e.g. input states and output reactions and 5) graphical-component nodes, representing components which do not contain any rxncon states but get regulated by the rxncon system. Figure 2.5 shows the visualisation of the regulatory layer of the insulin pathway as a rxncon species-reaction graph.

The edges are unidirectional and depict the information flow through the graph. However, in contrast to the Influence graph [146], the effect of the graphical-reaction and state nodes is strictly separated by different edge types. We consider two categories of edges: contingency edges and reaction edges. Each contingency type listed above has a separate edge. The edges are pointing from graphical-state nodes to either graphical-Boolean nodes or graphical-reaction nodes. Additionally, we have six edge types showing the effect of rxncon reactions on rxncon states (Table 2.5).

Table 2.5: Species-reaction graph edges.

Edge type	Description
produce	Pointing from a graphical-reaction node to graphical-state nodes, which get produced.
consumed	Pointing from a graphical-reaction node to graphical-state nodes, which get consumed.
synthesise	Pointing from a graphical-reaction node to graphical-state nodes in their neutral form belonging to a component which is synthesised by a rxncon reaction.
degrade	Pointing from a graphical-reaction node to graphical-state nodes belonging to a component which is degraded by a rxncon reaction.
maybe_degraded	Pointing from a graphical-reaction node to graphical-state nodes belonging to a component which is degraded by a rxncon reaction but these rxncon states are not explicitly mentioned in the contingencies or they are the complement of protected rxncon states.
source_state	Pointing from a graphical-reaction node to graphical-state nodes, which are required as sources for the rxncon reaction.

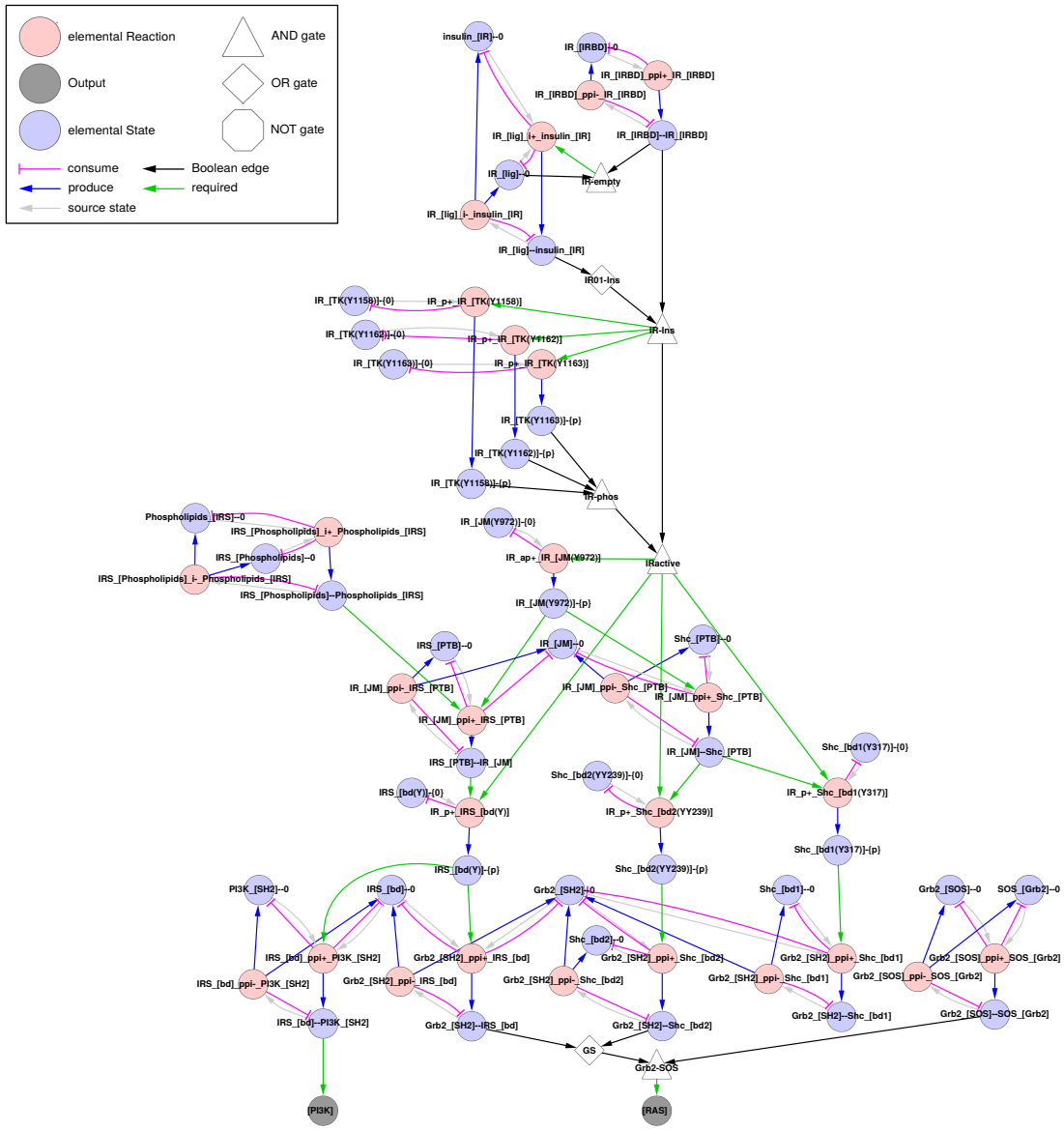


Figure 2.5: **Species-reaction graph of the insulin pathway**, shows a detailed view of the regulatory layer of the reconstructed network. The reaction edges depict which elemental reactions produce or consume which elemental states. Source state edges indicate the source of an elemental reaction. The contingency edges depict the regulatory relation between elemental states and elemental reactions. Elemental states defined within Boolean contingencies are connected through graphical-Boolean nodes. In this graph, we can follow the information flow from the receptor binding the ligand `IR_[lig]_i+_insulin_[IR]` through the pathway until it reaches the two reconstructed output reactions: `[PI3K]` and `[RAS]`.

A path from input to output is required but not sufficient for information transfer. Those graphs are useful to check whether it is possible to transmit the information through the network. In contrast to 'story' the regulatory graph visualises simultaneously all possible information paths through the network for a certain event. However, it is only meaningful to proceed to the model generation and evaluation step (Chapter 3) if the inputs and outputs are connected.

To demonstrate the scalability of the reconstruction language, we applied the method on the pheromone response pathway of Saccharomyces cerevisiae. We choose this pathway to benchmark rxncon due to the existence of a well annotated and detailed rule-based model [147]. We previously translated this model to rxncon [121] making it readily available for analysis. However, we adapted the model according to the changes in the newest version of the rxncon language. The rule-based model contains 229 rules with 200 parameters (166 unknown) which define a state space of over 200.000 distinct microstates based on 20 components [147, 148]. In comparison, the rxncon reconstruction of the yeast pheromone model contains 107 elemental reactions and 190 contingencies (Supplementary File SF3). We added undefined catalysts (UC) or catalysts for the ones we have no evidence for but are needed for a meaningful model. The reconstructed pheromone model has too many nodes and edges to be visualized with the species-reaction graph and is instead visualised with the regulatory graph, a sparse version of the species-reaction graph (Figure 2.6).

2.3.3 Regulatory graph

The regulatory graph is a sparse version of the species-reaction graph. In contrast to the species-reaction graph and the influence graph, the regulatory graph overviews the regulatory mechanisms of a reconstructed network by omitting the visualisation of neutral states, which reduces the complexity of the graph. Neutral states are implicitly encoded in graphical-component nodes, which are only visualised if there are degradation or synthesis reactions applied on the respective component. If a neutral state is given as Effector, this state will be included and visualised as a nested Boolean term of `OR`s of `NOT`s of its complements. Hence, events as well of feedbacks can be visualised in a condensed and clearer way compared to the species-reaction graph.

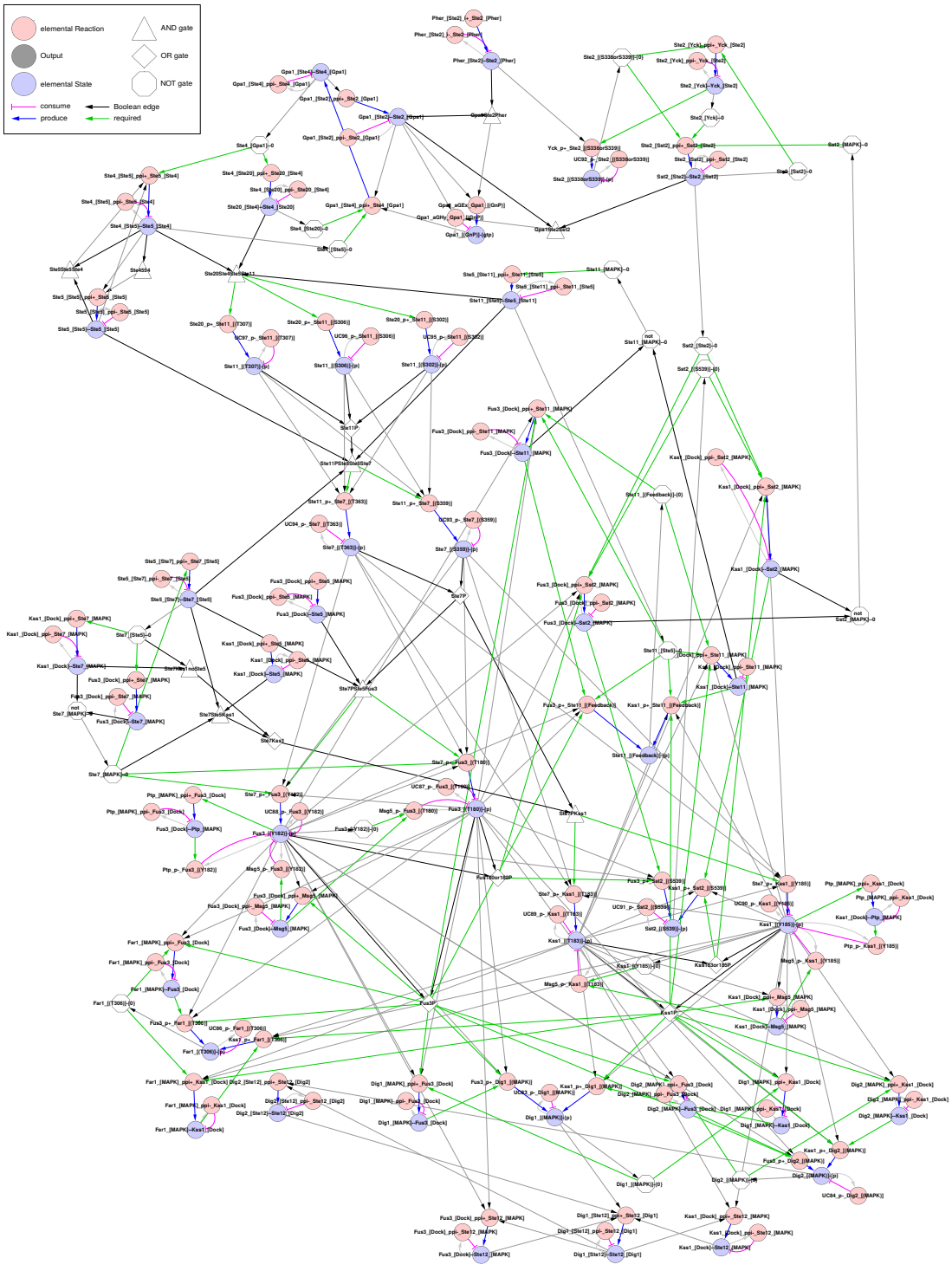


Figure 2.6: **Regulatory graph of the pheromone pathway**, shows a the regulatory layer of the reconstructed network. The reaction edges depict which elemental reactions produce or consume which elemental states. The contingency edges depict the regulatory relation between elemental states and elemental reactions. Elemental states defined within Boolean contingencies are connected through graphical-Boolean nodes. In this graph, we can follow the information flow from the receptor binding the ligand `Pher_[Ste2]_i+_Ste2_[Pher]` through the pathway until it reaches the regulatory reactions for Ste12.

2.4 Summary

- This chapter introduces the network reconstruction using the rxncon language.
- The rxncon language is based on context free grammar.
- We separate the reconstruction into decontextualised rxncon reaction and their contingencies.
- rxncon reactions produce, consume, synthesis or degrade rxncon states and describe the reaction layer of the rxncon system.
- Contingencies are contextual constraints on rxncon reactions, describing the regulatory layer.

CHAPTER 3

Network validation using Boolean modelling

This Chapter deals with the new bipartite Boolean model formalism based on the syntax and semantics of rxncon (Chapter 2). I show that we are able to predict meaningful biological functions on a system level and that this approach is a powerful tool to validate and simulate even large-scale signal transduction networks.

The reconstruction process described in Chapter 2 results in a mechanistic model, containing two layers of information: the regulatory layer, containing causal effects of elemental states on elemental reactions (contingency information) and the reaction layer, containing the rxncon reaction information of the rxncon system. After an initial reconstruction, the next step within the iterative reconstruction process is the validation of the network by reproducing an input-output relation of the network which can be validated with experimental data, e.g. phenotypical or functional data. Discrepancies between *in silico* prediction and *in vivo* data helps to discover knowledge gaps and to extend the model. This can be done in an iterative workflow - model building, model validation, gap finding and gap filling – to develop and debug the network [136].

For an *in silico* model validation we need a simple mathematical model that is able to predict the expected outcome with respect to a certain input signal. The translation of a network, representing a biological structure, into a quantitative model requires additional knowledge about rate laws and kinetic parameters as well as the integration of data from different levels in the case of large-scale models, e.g. whole cell models (see Chapter 1), which often need a large amount of experimental data. Furthermore, the high resolution of quantitative models can often not be satisfied with experimental data, leading to additional difficulties with respect to model building and low confidence in the analysis process. Hence, there is a need to simulate these systems at the qualitative level.

3.1 Boolean modelling in general

A quantitative model can be divided into a qualitative and a quantitative layer [133]. The qualitative layer requires information about the model topology, e.g. components, states, reactions and, in case of a signal-transduction model, their regulation, e.g. contingencies [149]. An advantage of simulating the qualitative layer is that it does not require kinetic parameters but enables a description of the qualitative dynamic behaviour. The community developed different methods to analyse the structure of a network. A common method is the Boolean model, first introduced in 1969 by Kauffmann [150] and used to simulate gene regulatory networks [150, 151].

Boolean models use logical operators (Table 3.1) to connect variables (literals), which can have two distinct logical values, 0 and 1.

Table 3.1: Table of logical operators

Sign	In words
$\wedge$	Logical AND / conjunction / intersection
$\vee$	Logical OR / disjunction / union
$\neg$	Logical NOT / complement

Literals that are connected by logical operators are called Boolean terms, which can be used to express any Boolean function and therefore, any logical expression. In rxncon, we use Boolean terms to describe biological complexes within the contingency term and call the literals of a Boolean Term *Effectors* (see Chapter 2). Each literal is described by a Boolean function, for which a single logical value is calculated. Different Boolean functions can be assigned as logically equivalent if we get the same output, using the same input, e.g. $\neg(A \wedge B) == \neg(A) \vee \neg(B)$ are equivalent as stated by the DeMorgan Theorem (Table 3.2).

Table 3.2: Table of logical equivalence, defined by the DeMorgan Theorem.

A	B	$\neg(A \wedge B)$	$\neg(A) \vee \neg(B)$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

The logical algebra defines complete classes of logical equivalent statements, so called normal forms. One of these normal forms is the disjunctive normal form (DNF), defining that disjunction terms only contain conjunction terms (equation 3.1).

$$\bigcup_i \bigcap_j (\neg) x_{ij} \tag{3.1}$$

The counter part of the DNF is the conjunctive normal form (CNF, equation 3.2).

$$\bigcap_i \bigcup_j (\neg) x_{ij} \tag{3.2}$$

The DNF and CNF are used later on to calculate the influence of degradation reaction nodes on the system and to describe Boolean components of the system, respectively (discussed 3.5.1).

A reconstructed network based on the Boolean logic can be represented as a Boolean network. This network is defined as a graph $G(V, E)$, containing nodes (Boolean nodes) $V = x_1, \ldots, x_n$ that represent biological molecules or global quantities and edges E between those Boolean nodes. The edges define Boolean update functions $E = f_1, \ldots, f_m$, representing the causality between different update steps (time points). Each Boolean node x_i has a Boolean state at time point t that can take two distinct logical values $x_i(t) \in \{0, 1\}$, representing the absence or presence of molecule properties, e.g. modifications or bindings.

The Boolean state representing a molecule x_i at time point $t + 1$ is determined by the Boolean function $f_i \in E$ that evaluates the Boolean states of Boolean nodes, influencing x_i at time point t. The change of the logical value from $x_i(t)$ to $x_i(t + 1)$ is called transition and can be defined by:

$$x_i(t + 1) = f(x_{j_{1(i)}}(t), x_{j_{2(i)}}(t), \ldots, x_{j_{k(i)}}(t)) \tag{3.3}$$

where $k(i)$ is the regulator of x_j and $j_{k(i)}$ is the mapping between different Boolean nodes at time point t influencing x. This can also be written as $x(t + 1) = f(x(t))$. The transition between nodes is deterministic: given a Boolean state $x_i(t)$, its successor Boolean state $x_i(t + 1)$ is unique.

All Boolean states at time t can be described by a Boolean state vector $S(t)$ (vector of Boolean states) of the network at time t: $S(t) = (x_1(t), x_2(t), \ldots, x_n(t))$. Since every Boolean state x_i can only take the logical values $\{0, 1\}$ at time point t, the dimensionality of the Boolean state space (number of all possible Boolean state vectors) is 2^n, where n is the number of Boolean nodes. The number of Boolean nodes in a model representing a biological system is finite, hence, the Boolean state space is finite. Amongst others, the Boolean state space consists of attractors and transient state vectors, leading to attractors. The point attractor or singleton attractor describes a Boolean state vector that cannot be left once it is reached, whereas the cyclic attractor describes Boolean state vectors that can be reached periodically during a simulation. However, the simulation of time within a Boolean model strongly depends on the way the nodes are updated during the simulation. The most simple Boolean simulation uses the synchronous update, which affects all Boolean nodes in each time step, whereas the asynchronous update only affects a subset of Boolean nodes. The selection of the affected subset depends on the selection method, e.g. random selection [152, 153]. In this thesis I use only the synchronous update for Boolean simulations.

3.2 Related work

A classical Boolean modelling approach is to model each protein as a node. Due to the simplicity and scalability of this approach Boolean modelling is frequently used to analyse the qualitative behaviour of signal-transduction networks [154, 155, 156]. However, modelling each protein as a single node is problematic, because many signalling components are differentially activated for different downstream mechanisms. The generic activation or inactivation through

the lack of mechanistic detailed information, e.g. crosstalk between pathways by a protein which is differentially modified, makes the classical Boolean approach not suitable to model signal-transduction networks.

To overcome this problem Boolean model formalisms were developed that are able to handle mechanistic information [144, 157, 158] (further reading [159]), which can be used for the detailed description of signalling events. One Boolean model approach combines rule-based and site-specific logical modelling and therefore, requires threshold parameters on top of a fully parametrised rule-based model, which makes this approach unsuitable for large-scale signal transduction networks [158]. An alternative Boolean model approach is the bipartite Boolean model that is able to distinguish between different downstream functions of biological components [144]. Therefore, the network is described in terms of elemental reactions (reaction nodes) and elemental states (Boolean state nodess), enabling an approximation of the underlying qualitative dynamics of the reconstructed network. For rxncon 2.0, we developed a new bipartite Boolean modelling formalism (bBM) that is a reinvention and only loosely related to the previous version of the bipartite Boolean model [144]. Now we are able to translate any rxncon network into a uniquely defined, executable bipartite Boolean model based on two generic update functions for state nodes and reaction nodes, which do not require further parametrisation or optimisation at the system level.

3.3 The bipartite Boolean model

The main objective of the bipartite Boolean model is to represent the qualitative layer of a rule-based model – a model representing the dynamics of the system without any kinetic parameters, relative concentration or rate laws. To keep the the bipartite nature of the rxncon network within the Boolean network, we separate the Boolean nodes into two sets, one set representing elemental reactions and output reactions, so called reaction nodes $\mathcal{R} = \{R_i\}$ with $i = 1, \ldots, N_R$ (N_R is the number of reaction nodes) and one set representing elemental states and input states, so called state nodes $\mathcal{S} = \{S_i\}$ with $i = 1, \ldots, N_S$ (N_S is the number of state nodes). Note, the following explanations are with respect to reaction and state nodes representing elemental states and elemental reactions if not stated otherwise. The handling of reaction nodes, representing output reactions and state nodes, representing input states will be discussed in detail later.

The Boolean update function for reaction and state nodes are based on two distinct sets of edges: transition edges, describing the influence of a reaction node on the logical value of a state node and contingency edges, describing the regulation of the logical value of a reaction node by contingencies (a set of state nodes). The two sets of Boolean nodes and the two sets of edges are used to describe the dependencies within the regulatory structure. However, we need additional assumptions to translate the regulatory structure into a bipartite Boolean model. The first assumption states that the logical value of a state node is derived from a local equilibrium motif (Figure 3.3), meaning that (in absence of degradation) the state nodes are active if the reactions producing the state nodes are active. The second assumption states that a state node is active if the rxncon state is abundant enough to be measured and therefore, functionally important for the system. This in turn implies that if the state node is not active, the rxncon state is not functionally important for the biological system. The third assumption states that any microstate (see Chapter 2) described by the intersection of elemental states is present in the Boolean system as long as the required individual state nodes are active. The

combination of assumptions one and two imply that a state node (in the presence of production and consumption and absence of synthesis and degradation reactions) can only be turned off if and only if the reaction node producing the state node is inhibited. Assumption three ensures that contingencies defined on a single molecule-level (as it is the case in rxncon systems, see Chapter 2), can be translated into systems-level quantities. All three assumptions enable the construction of generic update functions for state nodes that represent elemental states and reaction nodes that represent elemental reactions.

We have to translate the rxncon model into a set of reaction nodes and a set of state nodes to build the bipartite Boolean model. In addition to state nodes, representing elemental states, we enable the representation of state nodes which do not carry any further structure, e.g. modification or binding, so called generic component state nodes. The explicit modelling of generic component state nodes enables synthesis and degradation by generic proteins and the modelling of kinases as well as the regulation of such reaction nodes through the generic component state nodes. Since reaction nodes are representing elemental reactions, e.g. interaction or modification reactions, they fall into four distinct categories depending on the effect on their target states: production, consumption, synthesis and degradation and hence, they carry the information needed to create update functions. The different effects of rxncon reactions on rxncon states are defined by the skeleton rule - representing the reaction layer (Chapter 2). Additionally, elemental reactions are subject to contingencies - the regulatory layer. The reaction and the regulatory layer together with a few basic assumptions are sufficient to define the complete bipartite Boolean model.

3.4 The expected behaviour of a small reaction circuit

We designed two minimal circuits, one for interaction reactions and one for modification reactions, on the basis of already known biological systems to develop the update rules for the Boolean model. For both circuits we know the outcome to specific input signals, hence, we know the expected behaviour of the system. Those circuits can then be combined to more complex interaction systems, e.g. large-scale signal transduction networks. The first minimal circuit is designed to represent the expected behaviour of modification reactions and consists of two elemental states: a neutral elemental state and a phosphorylated elemental state. Additionally,

Table 3.3: Different effects of rxncon reactions on rxncon states within the minimal modification circuit.

Reaction	neutral state (A-{0})	modified rxncon state (A-{P})
Phosphorylation (p+)	Consumed	Produced
De-phosphorylation (p-)	Produced	Consumed
Synthesis (syn)	Synthesised	N/A
Degradation (deg)	Degraded	Degraded

we added four elemental reactions to the system: one phosphorylation reaction (p+), producing the phosphorylated rxncon state and consuming the neutral state; one de-phosphorylation reaction (p-), producing the neutral state and consuming the phosphorylated rxncon state; one

synthesis reaction, synthesising the neutral state; one degradation reaction, degrading both, the neutral and the phosphorylated rxncon state (Figure 3.1, Table 3.3).

The second minimal circuit is designed to represent the expected behaviour of an interaction reaction, consisting of three elemental states: two unbound neutral elemental states and one elemental interaction state. In addition, we added four elemental reactions to the system: one protein-protein interaction forward reaction (`ppi+`), producing the elemental interaction state and consuming both unbound neutral states; one protein-protein interaction reverse reaction (`ppi-`), producing both unbound neutral states and consuming the rxncon interaction state; one synthesis reaction synthesising one of the unbound neutral states; one degradation reaction degrading component A, contained in the unbound neutral state and the rxncon interaction state. In case of the degradation of the rxncon interaction state we expect that the rxncon interaction state gets consumed and the binding partner, which is not target of the degradation, gets released to the system (Figure 3.2, Table 3.4).

Table 3.4: Different effects of rxncon reactions on rxncon states within the minimal interaction circuit.

Reaction	neutral unbound state `A--0`	neutral unbound state `B--0`	rxncon interaction state `A--B`
ppi forward reaction (`ppi+`)	Consumed	Consumed	Produced
ppi reverse reaction (`ppi-`)	Produced	Produced	Consumed
Synthesis (`syn`)	Synthesised	N/A	N/A
Degradation (`deg`)	Degraded	N/A	Degraded

The different logical tables depend on the different literals in the update function, representing a local equilibrium. A system in absence of reaction nodes will have no transition of the logical value of the state nodes and hence, they will remain as initiated (Figure 3.1B). If we include different combinations of producing and consuming reactions, e.g. the phosphorylation and de-phosphorylation reaction nodes as in the first minimal circuit, we expect three different outcomes: 1) if only the phosphorylation reaction node is active, we expect the Boolean component to get into a fully phosphorylated Boolean state, i.e. the neutral state node disappears, 2) if only the de-phosphorylation reaction node is active we expect the Boolean component to get into a fully de-phosphorylated Boolean state, i.e. the phosphorylated state node disappears and 3) if both reaction nodes are active we expect both forms to be active in an equilibrium (Figure 3.3). For all these cases we require the Boolean component to be present. However, it is not important to know which state node of the Boolean component is available (Figure 3.1C). This behaviour also holds for the second minimal circuit.

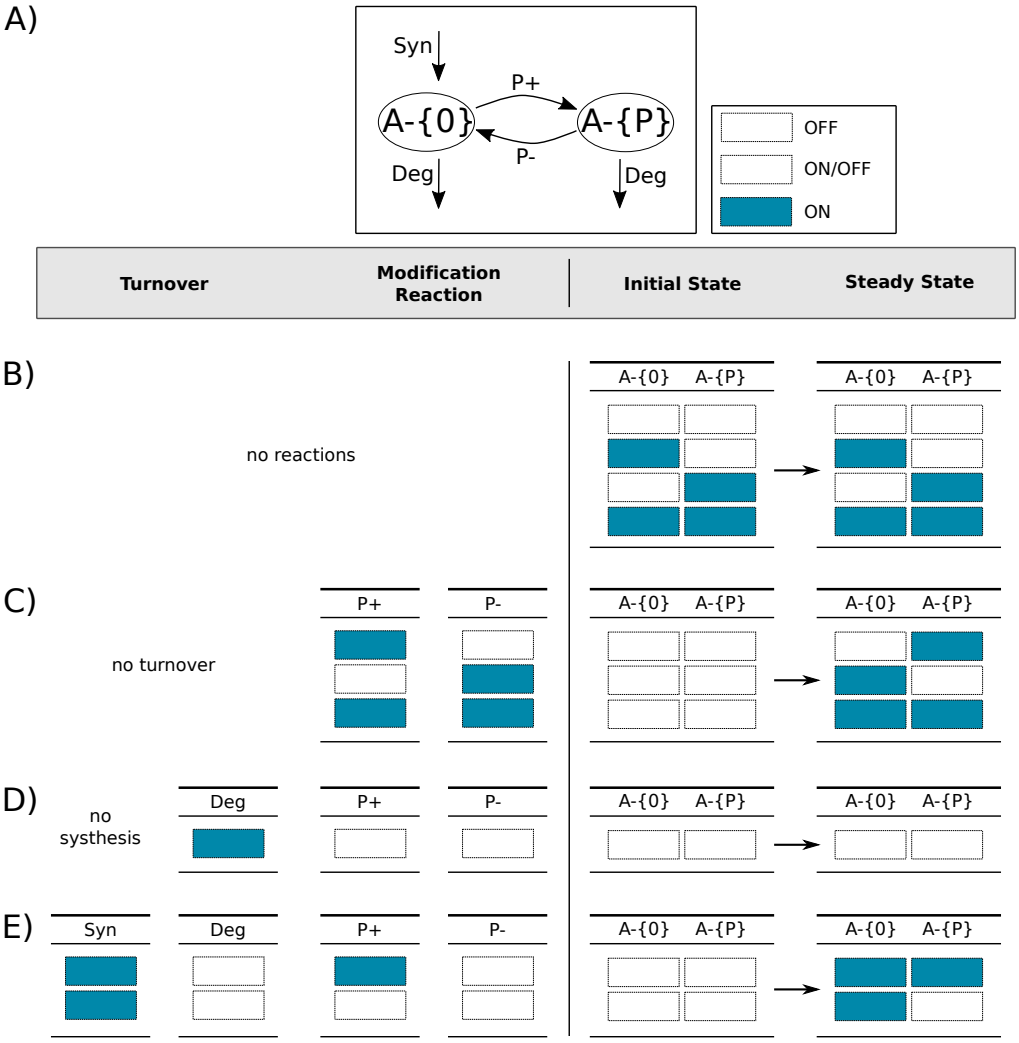


Figure 3.1: **Expected behaviour of minimal modification circuit.** Here, we show a minimal modification circuit with four reactions and two elemental states of a generic protein (A). A) The minimal modification circuit: A generic protein is synthesised (`syn`), degraded (`deg`), phosphorylated (`p+`) and de-phosphorylated (`p-`). B) In a Boolean system with inactive reaction nodes, we expect that the state nodes in steady state will remain the same as the initial state nodes. C) In a system with inactive protein turnover (`syn`, `deg`) but an active phosphorylation reaction node, we expect that the protein will be fully phosphorylated in steady state. Furthermore, we expect that the phosphorylated state node is inactive if only the de-phosphorylation reaction node is active. In this case, the protein will be fully de-phosphorylated in steady state. With both reactions active, we expect both state nodes to be present at equilibrium. For all these cases, we require the Boolean component to be present. D) In a system with an active degradation node but inactive synthesis node, neither of the two state nodes will be present regardless of the (de-)phosphorylation reactions and initial states. E) In a system with protein synthesis and phosphorylation, both state nodes will be active, regardless of the presence of the other reactions and the initial values. Without phosphorylation, only the unmodified state node will be active in the system, except if degradation and phosphorylation are inactive but the phosphorylated state node is present initially. As no reactions affect the phosphorylated protein, it will remain in the system (compare panel B).

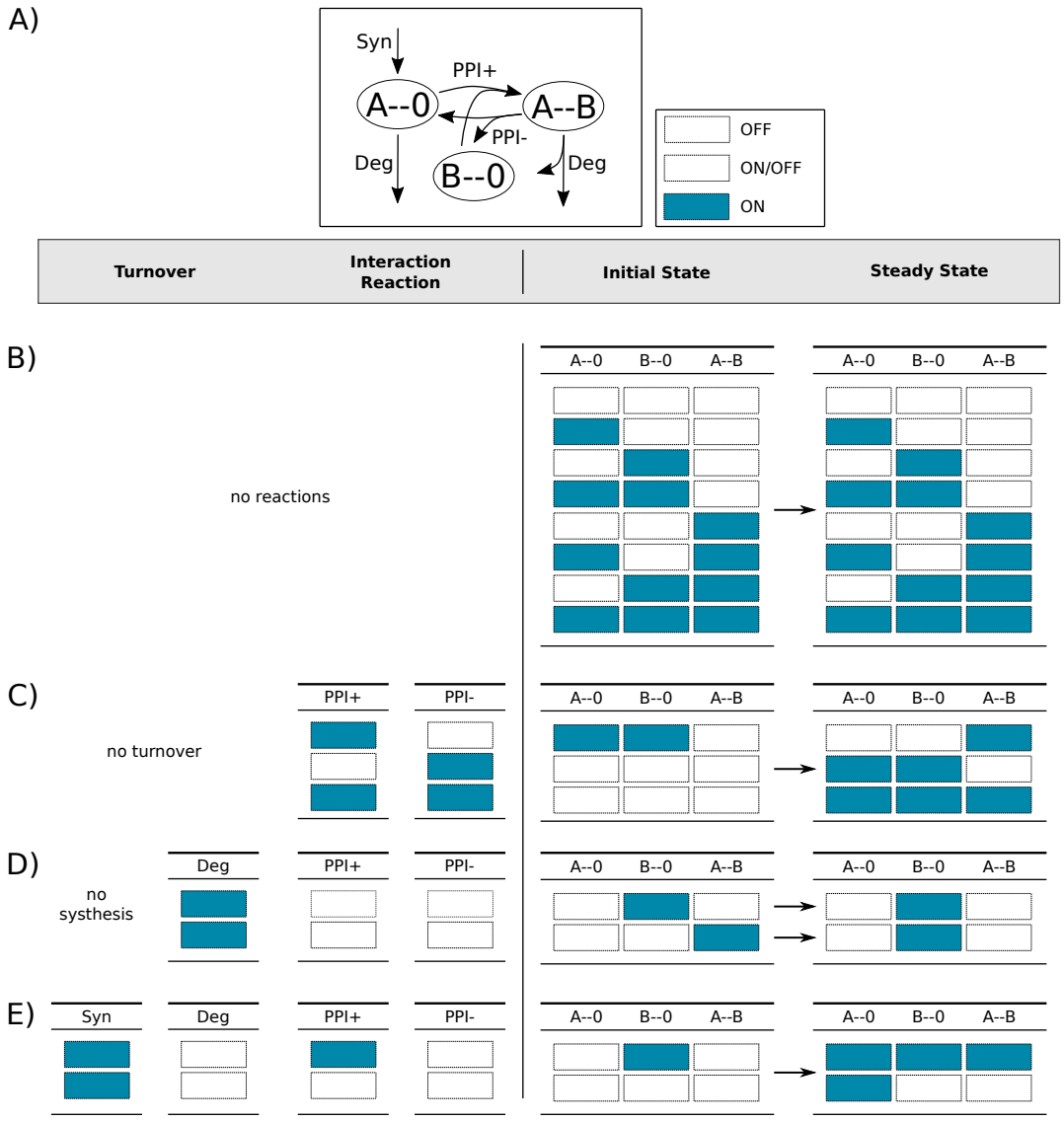


Figure 3.2: **Expected behaviour of minimal interaction circuit.** Here, we show a minimal interaction circuit with four reactions and three elemental states of two generic proteins (`A`,`B`). A) The minimal interaction circuit: A generic protein is synthesised (`syn`), degraded (`deg`) and undergoes an protein-protein forward (`ppi+`) and reverse (`ppi-`) interaction with another generic protein. B) In a Boolean system with inactive reaction nodes, we expect that the state nodes in steady state will remain the same as the initial state nodes. C) For the following cases, we assume that the respective Boolean components are present in the system. In a system with inactive protein turnover (`syn`, `deg`) but an active `ppi+` reaction node, the protein will be fully bound to the other protein in steady state. Furthermore, we expect that the interaction state node is inactive if only the `ppi-` reaction node is active. In this case, both proteins will be fully unbound in steady state. With both reactions active, we expect all three state nodes to be present at equilibrium. D) In a system with an active degradation node but inactive synthesis node, the unbound state node `A--0` and the interaction state node are inactive regardless of the ppi reactions and initial states. Only the unbound state node `B--0` will be active if the state node is present initially or the interaction state node is active, which will produce the state node `B--0` during degradation. E) In a system with protein synthesis and `ppi+`, the synthesised unbound state node and the interaction state node will be active, regardless of the presence of the other reactions and the initial values. Without `ppi+`, only the unbound state nodes will be active in the system, except if degradation and `ppi+` are inactive but the interaction state node is present initially. As no reactions affect the interaction protein, it will remain in the system (compare panel B).

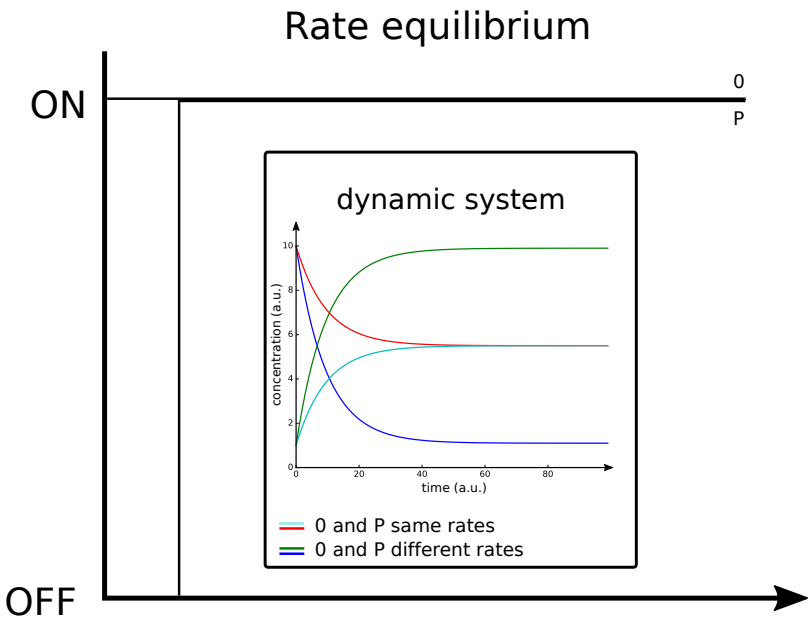


Figure 3.3: **Rate equilibrium comparison between quantitative and qualitative models.** The quantitative and a qualitative system have two different states: unphosphorylated (0) and phosphorylated (P). The inner graph shows a quantitative system, assuming Michaelis-Menten kinetics. I) The rate constants are identical, II) Phosphorylation is an order of magnitude faster. In both cases, the concentration of state P increases and the concentration of the state 0 decreases but both states will be present at equilibrium. To describe this system level behaviour within a qualitative model, the forward reaction has to be qualitatively dominant over the reverse reaction (from the perspective of each state), ensuring that both state nodes (0, P) are active in a reaction cycle where forward and reverse reaction nodes are active. This assumption is used in the bipartite Boolean model logic accordingly.

If we add a degradation reaction node to the modification circuit, we expect the complete Boolean component to be degraded regardless of the previously introduced reaction nodes and initial state nodes (Figure 3.1D). However, if we add a synthesis reaction node, we expect a similar behaviour as for phosphorylation and de-phosphorylation reaction nodes, meaning that we expect the state nodes that are synthesised and degraded to be present in an equilibrium. Additionally, we expect the modified state nodes of synthesised Boolean components to appear as soon as the phosphorylation reaction node is present even if the state nodes get degraded. However, if the synthesised state node appears but the phosphorylation reaction node is not active, we also don't expect to observe the phosphorylated state node as active. If there is no reaction node producing the phosphorylated state node, we have two possibilities: 1) the phosphorylated state node is present, hence, only stable in the absence of the de-phosphorylation and the degradation reaction node and 2) the phosphorylated state node is not present, in which case the state node will stay inactive. If all four reaction nodes are active, we expect both state nodes to be active, regardless of the activity of the de-phosphorylation and degradation

reaction node (Figure 3.1E).

The expected behaviour of the interaction circuit is similar to the modification circuit (Figure 3.2). However, within the interaction circuit an additional component is defined, which is not synthesised or degraded. We expect that the degradation of the rxncon interaction state leads to the release of the unbound neutral state of this component. Thereby, we make sure that the component, which is not target of the degradation, stays in the system.

3.5 Constructing the generic update functions of the bipartite Boolean model

The bipartite Boolean model is constructed in seven steps from a rxncon system. The first four steps involve the collection of rxncon system information (e.g. rxncon components, rxncon reactions and rxncon states) needed to construct the Boolean update functions and the last three steps involve the construction of the update functions for state nodes, reaction nodes and Boolean quantities. State nodes and reaction nodes are representing elemental states and elemental reactions, respectively. Therefore, they inherit the properties of the elemental states and elemental reactions they are representing (see Chapter 2).

3.5.1 Collecting rxncon system information

State nodes: The state nodes $\mathcal{S} = \{S_i\}$ with $i = 1, \ldots, N_S$, where N_S is the number of state nodes within the Boolean system, represent rxncon states. Elemental states carry the information that is transferred through the biological network and hence, they are part of the regulatory layer of the rxncon system.

Reaction nodes: The reaction nodes $\mathcal{R} = \{R_i\}$ with $i = 1, \ldots, N_R$, where N_R is the number of reaction nodes within the Boolean system, represents elemental reactions. Elemental reactions contain the information about which elemental state (represented by a state node) is produced, consumed, synthesised or degraded by an elemental reaction (represented by a reaction node) and hence, they provide the reaction layer of the rxncon system. This information is defined by the skeleton rule and integrated in the Boolean update function of the state node. Production, consumption, synthesis and degradation reactions can be seen as Boolean terms influencing the activity of a state node (Table 3.5).

$$p(R_i, S_j) = \begin{cases} 1 \text{ if } R_i \text{ produces } S_j \\ 0 \text{ otherwise} \end{cases}$$

$$c(R_i, S_j) = \begin{cases} 1 \text{ if } R_i \text{ consumes } S_j \\ 0 \text{ otherwise} \end{cases}$$

$$d(R_i, S_j) = \begin{cases} 1 \text{ if } R_i \text{ degrades } S_j \\ 0 \text{ otherwise} \end{cases}$$

$$s(R_i, S_j) = \begin{cases} 1 \text{ if } R_i \text{ synthesis } S_j \\ 0 \text{ otherwise} \end{cases}$$

Table 3.5: Definition of production, consumption, degradation and synthesis in a Boolean system.

Skeleton rule type	Meaning in Boolean system
Production	Transition between source state nodes and Boolean nodes. Evaluation of the Boolean term activates the targeted Boolean nodes.
Consumption	Transition between source state nodes and Boolean nodes. Evaluation of the Boolean term deactivates the targeted Boolean nodes.
Synthesis	The evaluation of the Boolean term leads to an activation of Boolean nodes representing neutral states and sharing the synthesised rxncon component
Degradation	The evaluation of the Boolean term leads to a deactivating of Boolean nodes sharing the degraded rxncon component

If we add degradation reaction nodes, we have to take care of their special role within a Boolean system. Degradation reaction nodes have both, a global and a local influence on the Boolean system. They can remove specific state nodes (local influence) or entire Boolean components (global influence) from the Boolean system. To integrate the degradation reaction node into the Boolean system, we have to first find the rxncon components influenced by the degradation reaction node and translate them into Boolean component expressions and second, check if the degradation rxncon reaction is regulated or not. If the degradation rxncon reaction is not regulated, all rxncon states sharing the same rxncon component should be degraded and therefore, all state nodes being part of the respective Boolean component expression are degraded. If the degradation rxncon reaction is regulated, the regulatory information has to be integrated. This information has to be considered on top of the skeleton rule, because we have to define which rxncon states are affected by the degradation rxncon reaction. Since the regulatory information is represented by contingencies, defined as Boolean terms of elemental states, we can calculate the disjunctive normal form (DNF, see 3.1) of the contingencies. The degradation reaction node will be affected differently and will have a different effect on the Boolean system by each of the conjunctive terms within the solution. Hence, we have to split the degradation reaction node for each conjunctive term and map the respective regulatory information of the conjunctive term on a Boolean expression of state nodes (Figure 3.4A). During this process, we have to exclude or include mutually exclusive state nodes with respect to the regulatory effect. If the degradation reaction node is inhibited by a state node, the mutually exclusive counterpart of the state node will be degraded but if the state node is required then the mutually exclusive counterpart will be protected from the degradation reaction node. However, in both cases, for regulated as well as for unregulated degradation reaction nodes, interaction state nodes have to be considered separately. If we degrade an interaction state node, the state node will be consumed but the binding partner (under the assumption that the state node does not represent a homo-dimer interaction state) will be released back to the Boolean system (Figure 3.4B). To ensure a controlled release of binding partners to the Boolean system, we have to include a

unique degradation for each interaction state node.
Therefore, we split the degradation reaction nodes in as many reactions nodes as there are interaction state nodes and assign each degradation reaction node to a certain interaction state node (Figure 3.4A).

We also have to add the information for synthesised state nodes to the respective synthesis reaction node. We distinguish between direct synthesis of neutral state nodes and indirect synthesis of non-neutral state nodes. A rxncon state is indirectly synthesised if the rxncon state is produced and the consumed rxncon state is synthesised. In case of multiple consumed rxncon states, e.g. rxncon interaction state, at least one of the consumed rxncon states has to be synthesised while the other rxncon states should be active (equation (3.4)).

$$\int(S_j) = \bigcup_{R_s \in S} R_s \cup \bigcup_{R_p \in P} R_p(\bigcap_{S_c \in C} (S_c \oplus_{R_g \in G} R_g) \tag{3.4}$$

where S: $R_s \in \mathcal{R}$ such that $s(R_s, S_j)$, P: $R_p \in \mathcal{R}$ such that $p(R_p, S_j)$, C: $S_c \in \mathcal{S}$ such that $c(R_p, S_c)$ and G: $R_g \in \mathcal{R}$ such that $s(R_g, S_c)$. Note, that for the indirect synthesis at least one of the source state nodes of the reaction node has to be synthesised.

Boolean components: The Boolean components $\mathcal{B} = \{B_i\}$ represent the rxncon components $\mathcal{X} = \{X_i\}$, being part of rxncon states or involved in a rxncon reaction. Information about the Boolean components are needed to construct the Boolean update function of the respective reaction node. To precisely handle the rxncon components we introduced a Boolean expression representing them

$$\xi(X_i) = (\bigcap_{S_a \in A} (S_a \bigcup_{S_b \in B} S_b)$$

, where $A : S_a \in \mathcal{S}$ such that $k(S_a, X_i)$ and $B : S_b \in \mathcal{S}$ such that $m(S_a, S_b)$ with:

$$k(S_a, X_i) = \begin{cases} 1 \text{ if } X_i \text{ is a component of } S_a \\ 0, \text{ otherwise} \end{cases}$$

$$m(S_a, S_b) = \begin{cases} 1 \text{ if } S_b \text{ is mutually exclusive with } S_a \\ 0, \text{ otherwise} \end{cases}$$

The Boolean expression contains all state nodes of rxncon states sharing a certain rxncon component such that the logical value of this Boolean expression indicates if a component is available in the Boolean system or not (this expression is further referred to as Boolean component expression or component expression). For instance, the minimal modification circuit consists of two mutually exclusive state nodes. The resulting Boolean component `A` is represented by $(A - \{p\} \vee A - \{0\})$. Hence, we are able to uniquely represent and evaluate the rxncon components in our system. To define all Boolean component expressions of the Boolean system we retrieve the information about which elemental state is part of which rxncon component.

Regulatory constraints: The regulatory constraints $\mathcal{C} = \{C_i\}$ with $i = 1, \ldots, N_C$, where N_C is the number of contingencies, are a mapping from the contingencies of a rxncon reaction (represented by the respective reaction node) to state nodes, influencing the reaction node through contingency edges. The function $\theta(R_i, C_i)$ calculates the contingencies regulating a

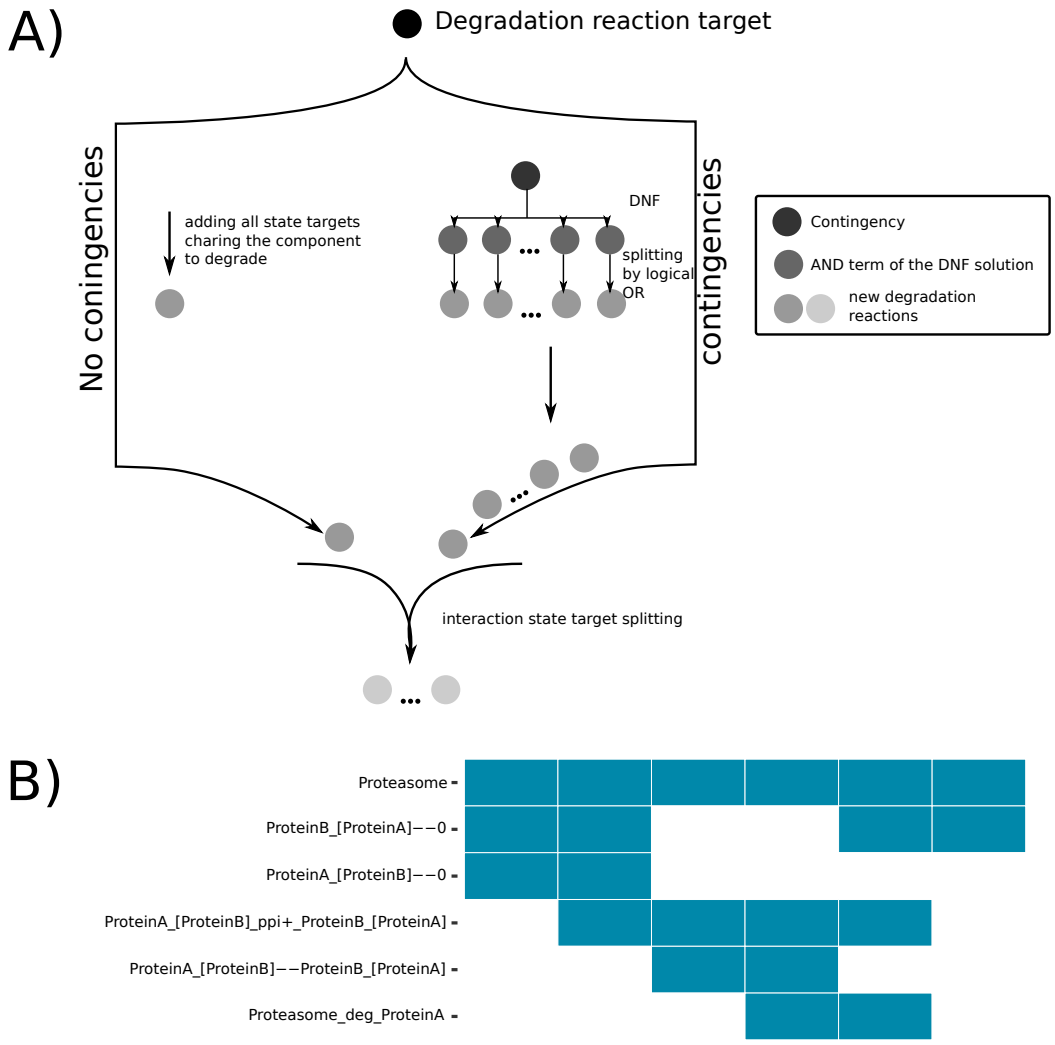


Figure 3.4: **The processing of degradation reactions.** The interpretation of degradation reactions depends on their regulation. A) shows two different ways to integrate degradation reactions into a Boolean system. If the degradation reaction is not regulated (left path), all state nodes containing the respective component will be considered for degradation. If the degradation reaction node is regulated, the regulatory information has to be considered on top of the skeleton rules. In a first step, a the disjunctive normal form (DNF) of the contingencies is calculated. For each conjunction term within the solution a degradation reaction node is created and the respective regulatory information of the conjunction term is added to this reaction node. In a last step, interaction states have to be considered for both cases, the regulated as well as the not regulated degradation reaction nodes. Since we have two interacting partners in an interaction state node, the degradation reactions will consume the state node but will release the partner, which is not degraded (in the case of homo-dimer interaction both partners will be degraded). For this step, the degradation reaction node has to be divided in as many reaction nodes as there are valid interaction state nodes for that particular degradation reaction. B) shows a simulation of the targeted degradation depending on an interaction state. As soon as the interaction state is created the degradation reaction turns on and degrades the interaction state. During this process the interaction partner is released in its neutral unbound form.

certain reaction:

$$\theta(R_i, C_i) = \begin{cases} 1 \text{ if } C_i \text{ is a contingency term for } R_i \\ 0, \text{ otherwise} \end{cases}$$

The contingency type is defined by the contingency term. However, quantitative contingencies do not fit the all-or-nothing principle of Boolean models. Hence, they are ignored or mapped to qualitative contingencies (discussed later). The information about regulatory constraints is needed to construct the Boolean update function of the respective reaction node.

After collecting all the information about state nodes, reaction nodes, Boolean components and regulations we can calculate the Boolean update functions according to the generic update functions. Note, that the update function for a reaction node only represents the regulatory layer of the rxncon system, hence, the source state nodes of a reaction node are not included here.

3.5.2 Update function

Update function for reaction nodes: The Boolean update functions for reaction nodes reflect the regulatory layer, directly corresponding to the reconstructed contingencies (see Chapter 2). Hence, all reaction nodes are independent of each other and can only influence and are only influenced by state nodes. The influence of reaction nodes on the system is controlled by two steps: the regulatory condition and the source condition. The regulatory condition defines the logical value of the reaction node. In a Boolean system without synthesis and degradation reaction nodes the logical value of a reaction node is True if all required state nodes are True and all inhibitory state nodes are False. If a reaction node has an actual effect on the Boolean system, it is regulated by the source condition, meaning that the effect depends on the available source state nodes, which is not a property of the regulatory layer. This in turn means that an active reaction node does not necessarily has an impact on the Boolean system. These control steps are needed due to both, technical and conceptual reasons: the update function for a reaction node should capture the contingency layer, which does not include the source state node information and the division into regulatory and source conditions avoids artefacts during the smoothing (discussed later). However, if we add synthesis and degradation reaction nodes to the Boolean system we need to include the Boolean components into the update function for reaction nodes.

$$\sum(R_i) = \bigcap_{X_a \in A} \xi(X_a) \bigcap_{C_j \in J} C_j \tag{3.5}$$

where A: $X_a \in \mathcal{X}$ such that $\tilde{k}(R_i, X_a)$ and J: $C_j \in \mathcal{C}$ such that $\theta(R_i, C_j)$ with:

$$\tilde{k}(R_i, X_a) = \begin{cases} 1 \text{ if } X_a \text{ is a component of a reactant of } R_i \\ 0, \text{ otherwise} \end{cases}$$

In this case, the logical value of a reaction node is True if all Boolean components, representing the the respective reactants and all required state nodes are True, and all inhibiting state nodes are False (Equation (3.5)). Taken together, the activity of reaction nodes is highly controlled by state nodes, reflecting the regulatory layer of the reconstructed system.

Update function for state nodes: The update function for state nodes defines the effect of reaction nodes on the logical value of the state nodes and therefore, the effect of reaction nodes on the Boolean system, representing the reaction layer. The effect of a reaction node on a specific state node depends on the type of the state node, e.g. a non-modified state node gets consumed by a modification reaction, whereas a modified state node is produced by a modification reaction (Table 3.5). The effect on the Boolean system depends on the available source state nodes i.e. a reaction node has only an effect on the Boolean system if the reaction node and all its source state nodes are True. This additional condition avoids ghost activations of state nodes, e.g. reactions can stay active, even if the source state nodes are not available anymore. However, a state node that is not subject to any reaction node retains its current logical value. If reaction nodes are influencing the state node, the logical value is assembled with a certain hierarchy of the reaction nodes: synthesis reaction nodes are dominant over degradation reaction nodes that are dominant over productions reaction nodes that are dominant over consumption reaction nodes (Figure 3.2 and 3.1). If we assume a Boolean system with one producing and one consuming reaction node but no synthesis and no degradation reaction node, the logical value of a state node will be evaluated to True iff: 1) the production reaction node is True and all its source state nodes are True 2) the state node is True and either the consuming reaction node is False or one of its source state nodes is False (Equation 3.6).

$$\sum(S_i, n) = \int(S_i) \cup \bigcap_{R_d \in D} (\neg R_d \bigcap_{X_a \in A} \xi(X_a)) \cap \{\pi_n(S_i) \cup S_i \cap \neg\kappa_n(S_i)\} \tag{3.6}$$

where D: $R_d \in \mathcal{R}$ such that $d(R_d, S_i)$; A: $X_a \in \mathcal{X}$ such that $k(S_i, X_a)$. The argument n in $\sum(S_i, n)$ is a indicator of the recursion depth for this equation. In the basic case $n = 0$, which leads to a consumption part $\kappa_0(S_i)$ of the equation as

$$\kappa_0(S_i) = \bigcup_{R_l \in L} R_l \bigcap_{S_m \in M} (S_m \bigcap_{R_b \in B} \neg R_b)$$

where L: $R_l \in \mathcal{R}$ such that $c(R_l, S_i)$, M: $S_m \in \mathcal{S}$ such that $c(R_l, S_m)$ and B: $R_b \in \mathcal{R}$ such that $d(R_b, S_m)$ and a production part $\pi_0(S_i)$ as

$$\pi_0(S_i) = \bigcup_{R_j \in J} R_j \bigcap_{S_k \in K} (S_k \bigcup_{R_a \in A} \neg R_a)$$

where J: $R_j \in \mathcal{R}$ such that $p(R_j, S_i)$, K: $S_k \in \mathcal{S}$ such that $c(S_k, R_j)$ and A: $R_a \in \mathcal{R}$ such that $d(S_k, R_a)$. Note, that a state node, e.g. an unbound protein node is produced if the binding partner within the respective interaction node is degraded. Hence, the degradation reaction term R_a is only applied in the production part π_n if the reaction node R_j producing the state node S_i is not equal to the degradation node R_a.

Adding synthesis and degradation reaction nodes to this Boolean system will have a different effect on the update functions of neutral and non-neutral state nodes (e.g. modified or bound state nodes). Neutral state nodes are the only state nodes, which are synthesised by default. Hence, in a Boolean system with active degradation nodes, a neutral state node is always active if the respective synthesis reaction node is active, whereas a non-neutral state node requires its indirect synthesis to overrule the influence of the degradation reaction node. However, even if the state update functions are more complex than the reaction update functions, the knowledge about the role and regulation of each reaction node on the state nodes is sufficient to create a

complete system of update functions.

Update function for global quantities: During the reconstruction process we give the possibility to define phenotypes or other global quantities via global output reactions, represented as global reaction nodes (or output reaction nodes) and global input states represented as global state node (or input state nodes). These global statements are handled differently since they are neither influenced by elemental reactions nor do they produce, consume, synthesis or degrade elemental states. They can influence the Boolean system in three different ways. First, we have a global state node influencing a reaction node and no global reaction node. In this case the global state node is completely decoupled from other state nodes of the Boolean system and the update function of a reaction node will be extended by this global state node according to its regulatory effect. Second, we have a global reaction node and no global state node. Hence, there are no contingencies that contain the global statement as Effector and therefore, it has no effect on other nodes of the system. Third, we have both, a global state node, which influences a reaction node and a global reaction node. Here, we define that the global reaction node 'produces' the global state node if both have the same name, meaning that the update function of the global reaction node is assigned to the global state node. Degradation reaction nodes are handled differently due to their special influence on the Boolean system. If a degradation reaction node is influenced only by a global state node, the Boolean component will be degraded by this reaction node. If both, a state node and a global state node are available for activating the degradation reaction node, only the state node gets depleted.

3.6 Simulation of the minimal circuits

The Boolean update functions for reaction and state nodes define the bipartite Boolean model completely but we have to test if the update functions capture the logic described at the beginning. Therefore, we generated 64 models corresponding to the minimal modification circuit from above (Figure 3.1A) as well as 144 models corresponding to the minimal interaction circuit (Figure 3.2) and simulated them using BoolNet [160]. The resulting attractor states are visualised in Figure (Figure 3.5C and Figure 3.6C).

The models behave as expected with three exceptions. Within the minimal modification circuit, in the presence of the phosphorylation and de-phosphorylation reaction nodes but absence of the synthesis and degradation reaction nodes, we observe an oscillatory behaviour if only one of the two state nodes is initiated as active (Figure 3.5C). A similar behaviour can be observed within the minimal interaction circuit (Figure 3.6C).

If the forward and reverse protein-protein interaction reaction nodes are active in absence of synthesis and degradation, we observe an oscillatory behaviour as long as one of the initial states is inactive. In presence of synthesis and both, forward and reverse protein-protein interactions, the oscillatory behaviour is only observed if one of the not synthesised initial states is inactive. This can be explained by the periodic source state node consumption, due to the constitutive activation of the production and consumption reaction nodes. Hence, the activity of the reaction nodes alternate, triggering an out-of-phase oscillation of the state nodes. The oscillatory behaviour disappears if all state nodes are initiated with True or, in the case of the minimal modification circuit, a synthesis reaction node constitutively activates the required source state node. These oscillations (also referred to as trivial oscillations) are appropriate for models on a single molecule-level and consistent with the definition of the rxncon system on a

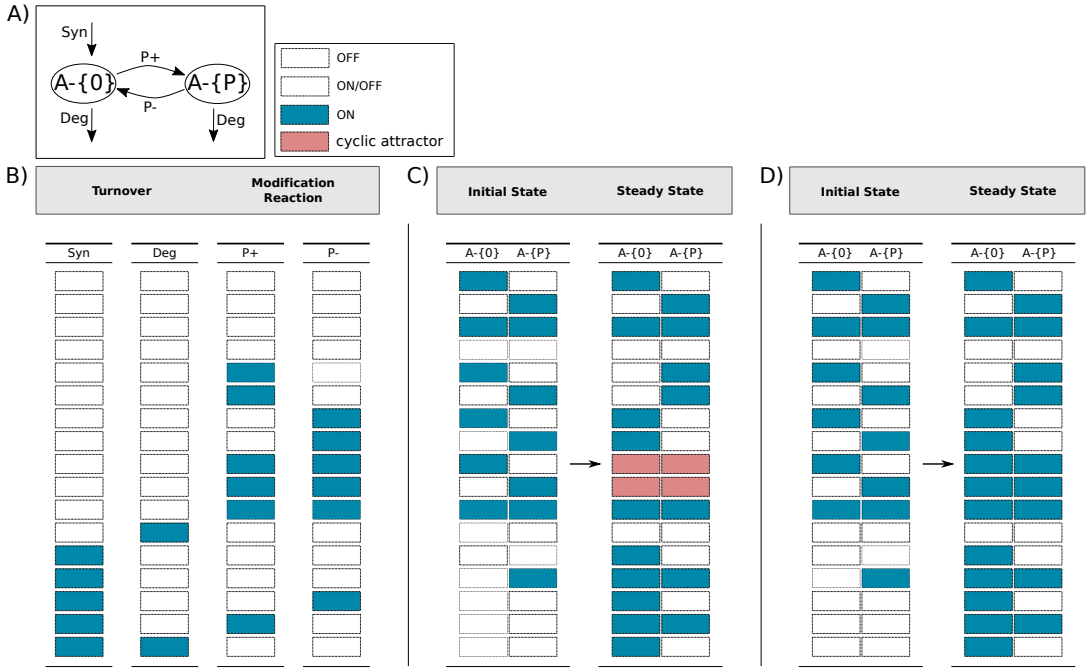


Figure 3.5: **Simulation of the minimal modification circuit.** A) For testing the bipartite Boolean logic the minimal modification circuitry with 2^2 (4) possible initial states was implemented and simulated. B) We build a model for each of the 4^2 (16) possible reaction configurations. C) Using the basic update functions, two state nodes oscillate out-of-phase between TRUE and FALSE (red) if both, phosphorylation and dephosphorylation reactions, are active and only one state node is initiated as TRUE. D) Using the source state smoothed update functions the oscillatory effect is avoided.

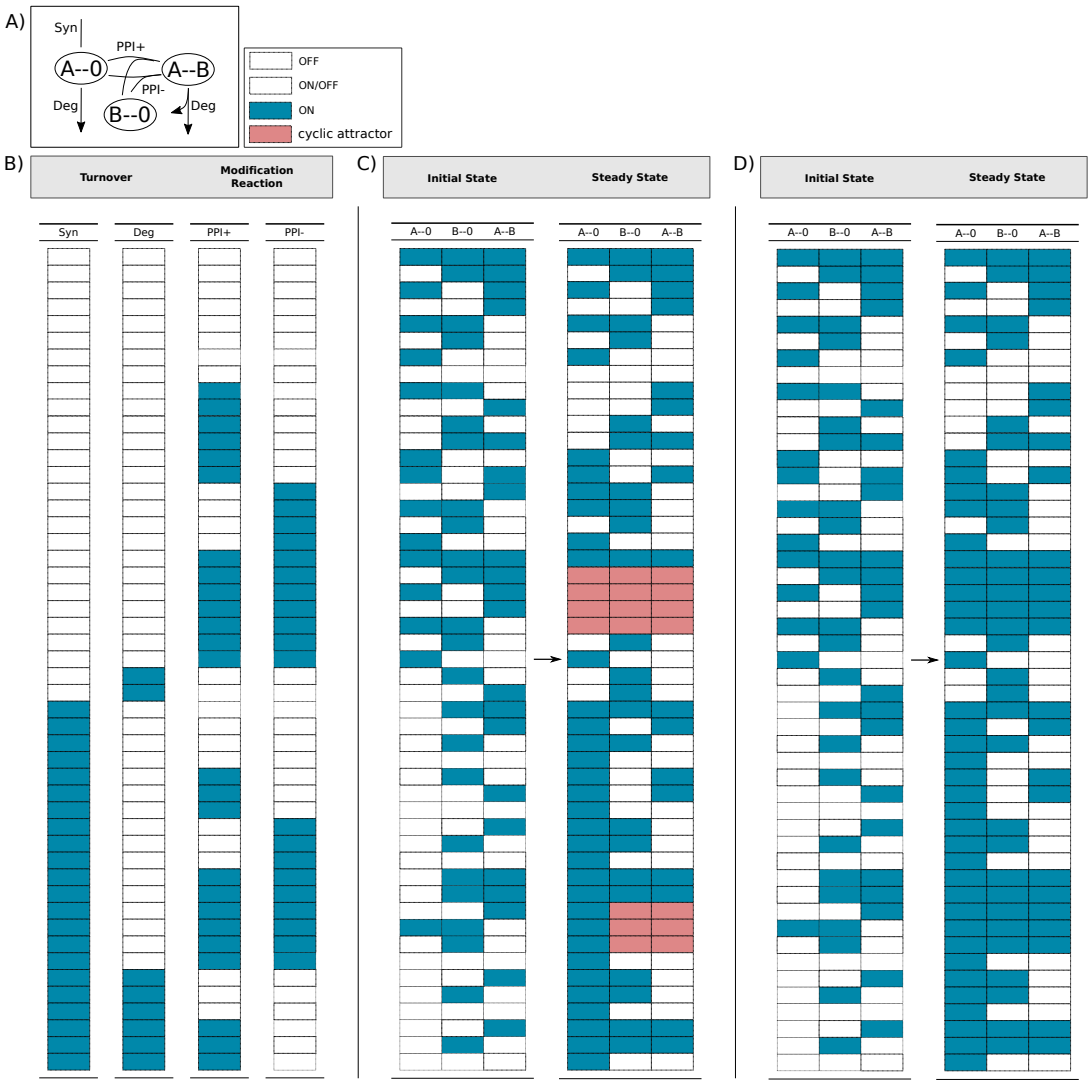


Figure 3.6: **Simulation of the minimal interaction circuit.** A) For testing the bipartite Boolean logic the minimal interaction circuitry with 3^2 (8) possible initial states was implemented and simulated. B) We build a model for each of the 4^2 (16) possible reaction configurations. C) Using the basic update functions, the state nodes oscillate out-of-phase (red) if both, forward and reverse protein-protein interaction reactions, are active in absence of degradation and if only one (not synthesised) state node is initiated as False. D) Using the source state smoothed update functions the oscillatory effect is avoided.

single-molecule level (Chapter 2) but not for models described on a system level. Nevertheless, we were able to reproduce the expected behaviour of 62 out of 64 minimal modification circuit models (Figure 3.5) as well as of 136 out of 144 minimal interaction circuit models(Figure 3.6), using our defined generic update functions and our system assumptions.

3.7 Single cell bipartite Boolean model

We adapted the bipartite Boolean model logic to avoid trivial oscillations and to capture single cell behaviour of signal-transduction networks. Trivial source state node oscillations within reaction node cycles are a consequence of state nodes (elemental reactions) that are mutually exclusive on a single-molecule level but simultaneously present at the system level. To avoid those trivial oscillations, we introduce a smoothing strategy – time and molecule smoothing, looking one time step ahead. The smoothing is based on the local equilibrium assumption, stating that if a forward and a reverse reaction node is active in a two-state node system both state nodes should be there at equilibrium. Within a qualitative modelling approach the equilibrium can be achieved by assuming a dominance of production reaction nodes over consumption reaction nodes and widening the window in which we check if a source state node for a respective reaction node is available. The applied smoothing changes the source conditions of reaction nodes. Now reaction nodes have an influence on the Boolean system if the required source state nodes are active or if their update function evaluates to True.

$$\pi_1(S_i) = \bigcup_{R_j \in J} R_j \bigcap_{S_k \in K} (S_k \bigcup_{R_a \in A} \neg R_a) \cup \sum(S_k, 0)$$

where J: $R_j \in \mathcal{R}$ such that $p(R_j, S_i)$, K: $S_k \in \mathcal{S}$ such that $c(S_k, R_j)$ and A: $R_a \in \mathcal{R}$ such that $d(S_k, R_a)$. Note, that the n for the update function for S_i is increased to 1 and that the n for the update function for S_k is 0.

This ensures that mutually exclusive state nodes on a single molecule-level can be active simultaneously, simulating a system level. If the molecule number is high enough and both rxncon reactions. creating mutually exclusive rxncon states are active, the rxncon states are present simultaneously in different biological molecules at any given time. However, this does not hold if we consider only few biological molecules and low reaction rates. For those cases, the rxncon reaction can be considered as functionally off. To test our model adaptation, we recreated all 64 models of the minimal modification circuit as well as all 144 models of the minimal interaction circuit with the smoothing logic and repeated the simulations. If we compare the results with the previous simulation (Figure 3.5D and Figure 3.6D), we can see that the oscillatory behaviour disappeared in both minimal circuits, using the smoothing. Hence, our simulation results match the initial expected behaviour of the minimal circuits.

3.8 Application on the Hog pathway

To test the bipartite Boolean model on a biological example, we applied the Boolean model to a linear pathway. Therefore, we reconstructed a simplified version of the Hog MAP kinase pathway from *Saccharomyces cerevisiae* (take from [144]), as a rxncon 2.0 model (Supplementary File SF4.1). We used this model to generate the bipartite Boolean model with the generic update functions with smoothing (Supplementary File SF4.2). As discussed above, we expect the output

to be responsive to the input. This can either be a positive response (activator) or negative response (inhibitor). In this pathway, we have an activator at the beginning of the signalling pathway, ensuring that the pathway is not activated if the input signal (Low Turgor) is not given. As output signal we want to observe the phosphorylated Hot1. Hence, we expect that the logical value of the state node `Hot1-p` gets active if the the global state node `[Turgor]` is inactive, otherwise not.

The model contains 28 reaction and state nodes, which results in a Boolean state space of 2^{28} possible initial Boolean states, which are too many possible initial Boolean states to test. Hence, we decided to use our generic but artificial initial conditions for the simulation, meaning that all state node representing neutral elemental states and all generic components (components without elemental states) are initialised with True, all other Boolean nodes are initialised with False. We run the model until it reaches an attractor, its own natural 'off-state', which is a point attractor with an inactive output signal. Now, we use this attractor but change the input state node Turgor from False to True and simulate the model again to observe the response from the output on the input. The model ends up in a point attractor where the output is ON, as expected. We repeat the process until the model returns to an attractor we have already seen (Figure 3.7B).

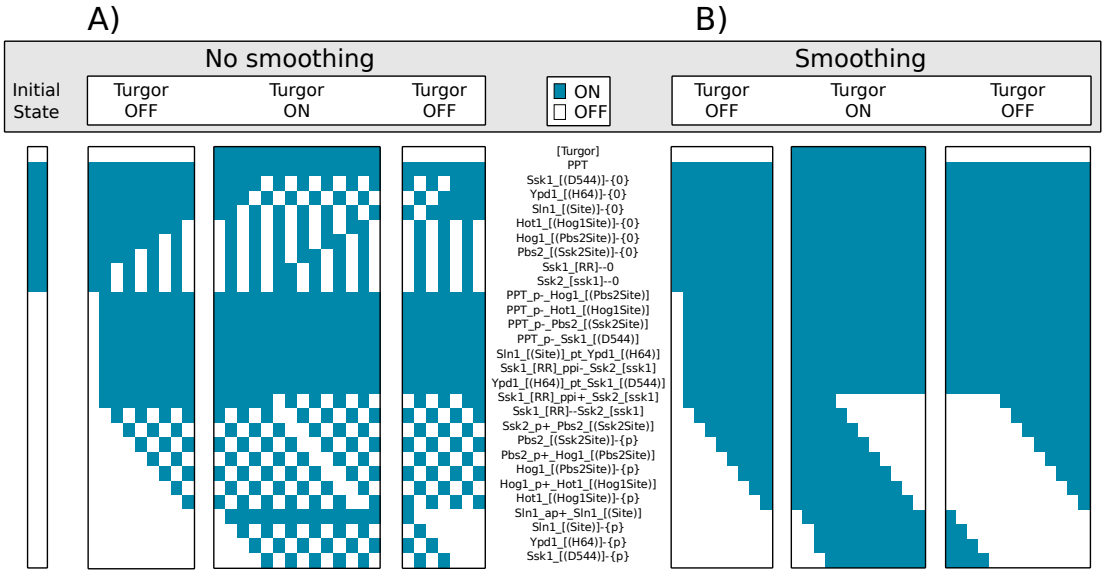


Figure 3.7: **The bipartite Boolean model simulation of a linear version of the Hog pathway.** The model is initialised with the default assumptions on the initial state. The model is simulated until it reaches an attractor (first Turgor OFF trajectory). The Hog pathway is activated afterwards by turning [Turgor] ON and the model is simulated again until an attractor is reached (Turgor ON trajectory). Next, [Turgor] is turned OFF again and the model is simulated until an attractor is reached. A) Simulation of the linear Hog pathway without smoothing. The signal goes through the pathway but due to the oscillatory behaviour the Boolean system is not able to reach point attractors. B) The simulation of a linear reconstruction of the Hog pathway with source state smoothing of the update functions. The model responds to the input as expected.

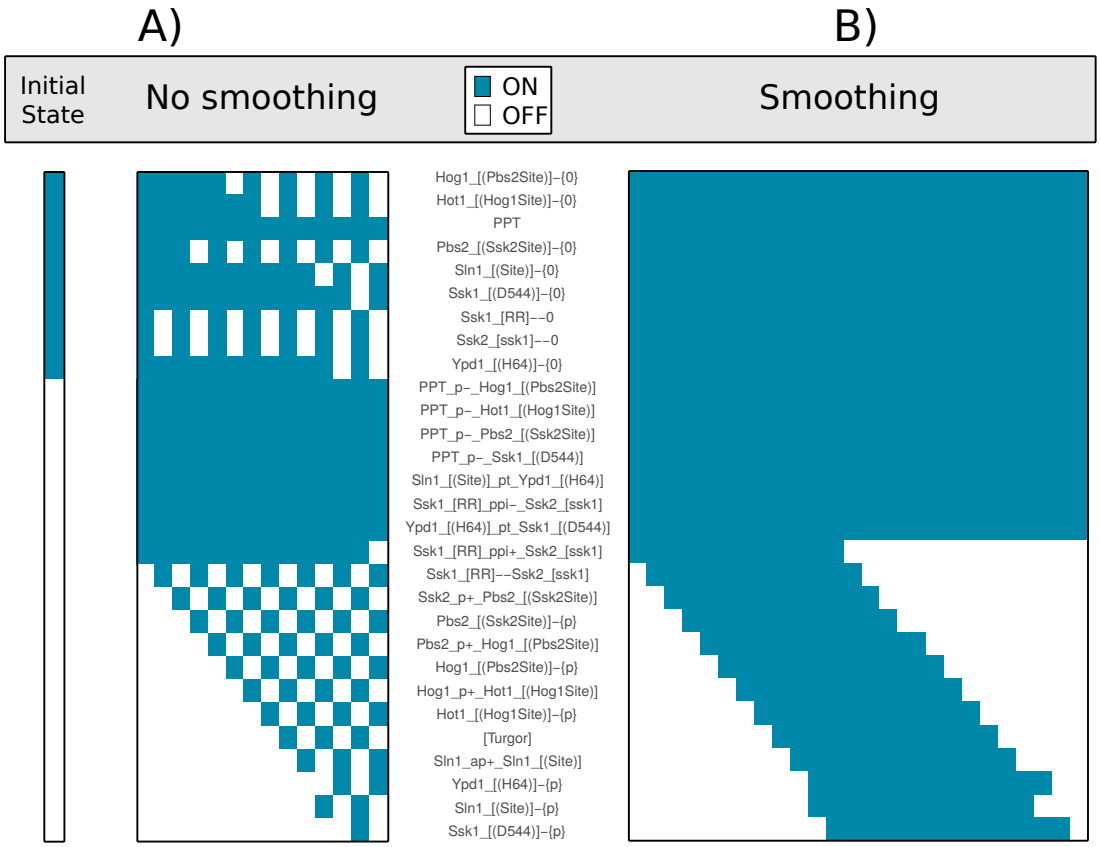


Figure 3.8: **The bipartite Boolean model simulation of a cyclic version of the Hog pathway.** A) Simulation of the cyclic Hog model without smoothing. The simulation shows a highly oscillatory behaviour, resulting in the loose of the expected systems behaviour. B) We extended the Hog model with a feedback loop, where activation of the pathway leads to increased turgor (via phosphorylated Hot1). This is a simplification of an adaptive response through increased glycerol production and retention, which increases turgor. The model is initialised with the default initial states. The system oscillates as expected.

We observe the expected response to the change in Turgor. Sln1 stays active if there is no osmotic stress to ensure that the downstream MAPK cascade does not get activated and we don't observe any trivial oscillations during the simulation. To compare the smoothed model to our first non-smoothed model we repeat the simulation with the non-smoothed logic (Figure 3.7A). We can see that the signal is passing the network despite trivial oscillations but the Boolean system does not converge into a point attractor as for the smoothed model. This leads to a more complex analysis and interpretation of the model. There are three blocks in the heatmap. The first block contains the initial neutral states that remain active because the reaction nodes producing them are considered as unregulated, which could be due to experimental bias (discussed below). The second block contains a block of reactions that turn on directly and stay active. These constitutive reaction nodes are either unregulated (de-phosphorylation reaction nodes) or have their source state nodes constitutively available (e.g. phosphotransfer from Sln1 to Ypd1). The third block turns on and off in response to the signal, which transmits the information through the Boolean network. Hence, we demonstrated, that the non-trivial generic update functions defined for isolated reaction nodes are sufficient to convert a rxncon network into a complete bipartite Boolean model. The functionality of a signal-transduction network is accurately predicted on a system level and does not need any further parametrisation. Taken together, the generic update functions are able to map any given rxncon network on a unique Boolean model that predicts system level function and is able to reproduce biological findings.

Next, we repeated the analysis with a cyclic pathway using our smoothing function. As described at the beginning of this work, the Hog pathway is a homeostatic pathway that maintains proper turgor pressure through Hot1-p, which inhibits the input signal through a physiological feedback loop [161]. To retrieve a cyclic model from our previous linear Hog model, we linked the most downstream component (Hot1-p) to the input, which will turn the pathway off (Supplementary File SF1, SF5). Hence, we expect that under low Turgor pressure (Turgor off = osmotic stress) Hot1 gets activated by phosphorylation, which leads to an increase of Turgor (through accumulation of glycerol), shutting down the Hog pathway. Upon re-inactivation of Hot1 Turgor pressure decreases. Hence, we expect an oscillatory behaviour of the system.

We initialise the model with the generic Boolean start state vector as for the linear model. The simulation shows a periodic behaviour of activating and deactivating state and reaction nodes. This periodicity is similar to what we observed if we change the input manually (Figure 3.8B). Hence, we are fully capable of predicting biological relevant oscillations. For comparison, we simulate the cyclic Hog model without source state smoothing (Figure 3.8A), resulting in a highly oscillating Boolean system. This shows, that the application of the smoothing within the generic functions of the bipartite Boolean model facilitates its analysis and improves the interpretability of the results. Hence, the bipartite Boolean model logic generates Boolean models which can predict system level functions of both, linear and cyclic systems.

3.9 Application on the pheromone response pathway

To demonstrate the scalability of the bipartite Boolean model, we applied the logic on the pheromone response pathway of *Saccharomyces cerevisiae* (for more detail see Chapter 1 & 2). As discussed, we choose this pathway due to its excellent annotation and detailed rule-based model by Thomson et al. [147, 148]. We simulated the pheromone bipartite Boolean model, using the standardised simulation workflow that we used for the Hog pathway. The reconstruction

contains 163 elemental reactions corresponding to 103 elemental states. The reconstruction was translated into a bipartite Boolean model with smoothed update functions, which results in 386 Boolean nodes, of which 208 Boolean nodes are reaction nodes and 178 Boolean nodes are state nodes. The higher number of reaction nodes compared to the elemental reactions, results from the contingency and interaction state effect on degradation reaction nodes as well as the split of bidirectional reactions into forward and reverse reactions, e.g. for interaction reaction nodes. In addition, we added generic component nodes for the undefined catalysts or synthesised generic components, e.g. Ste7mRNA, which increased the number of nodes but did not affect the complexity of the model. However, this results in a total Boolean state space of 2^{386} possible initial Boolean states. Hence, to simulate the model we use the default initial Boolean states as for the previous simulations (all neutral state nodes and all generic component nodes are True, all other Boolean nodes are False). The model finds its natural 'off-state'. However, the pathway did not behave as expected due to different quantitative effects, which were included during the reconstruction process. Through the all-or-nothing principle of Boolean models the quantitative information cannot be considered. Hence, the information has to be translated into qualitative information or will be ignored. We decided to turn a minimal set of quantitative contingencies (k+/-) that were ignored in the generation of the Boolean model, into qualitative contingencies (!/x). Therefore, 3 out of 91 quantitative contingency statements had to be changed. To realise the information transfer (Table 3.6) the left quantitative contingencies are ignored during the translation process (Supplementary File SF6, SF7).

Table 3.6: Pheromone model adaptation for bipartite Boolean simulation.

Target	Contingency type	Effector
Gpa1_[Ste4]_ppi_Ste4_[Gpa1]	x	Gpa1_[(GnP)]-{GTP}
Gpa1_aGEx_Gpa1_[(GnP)]	!	<Gpa1Ste2Pher>
<Gpa1Ste2Pher>	AND	Gpa1_[Ste2]--Ste2_[Gpa1]
<Gpa1Ste2Pher>	AND	Pher_[Ste2]--Ste2_[Pher]
Gpa1_aGHy_Gpa1_[(GnP)]	!	<Gpa1Ste2Sst2>
<Gpa1Ste2Sst2>	AND	Ste2_[Sst2]--Sst2_[Ste2]
<Gpa1Ste2Sst2>	AND	Gpa1_[Ste2]--Ste2_[Gpa1]

The simulation started from from the natural 'off-state'. We iteratively switched the input signal (pheromone), through its synthesis reaction regulated by UC99, to True and False (Figure 3.9). The updated version of the pheromone model behaves as expected, demonstrating that a simpler model is able to explain the functionality of this pathway.

In summary, the bipartite Boolean model logic is scalable and able to simulate systems which are too large to be simulated by quantitative methods, e.g. rule-based modelling. The generic update functions are able to map a rxncon reconstruction on a unique bipartite Boolean model that predicts the system level functionality and dynamics of the reconstructed model. Hence, the bipartite Boolean model is a suitable approach for validation and simulation of large-scale networks.

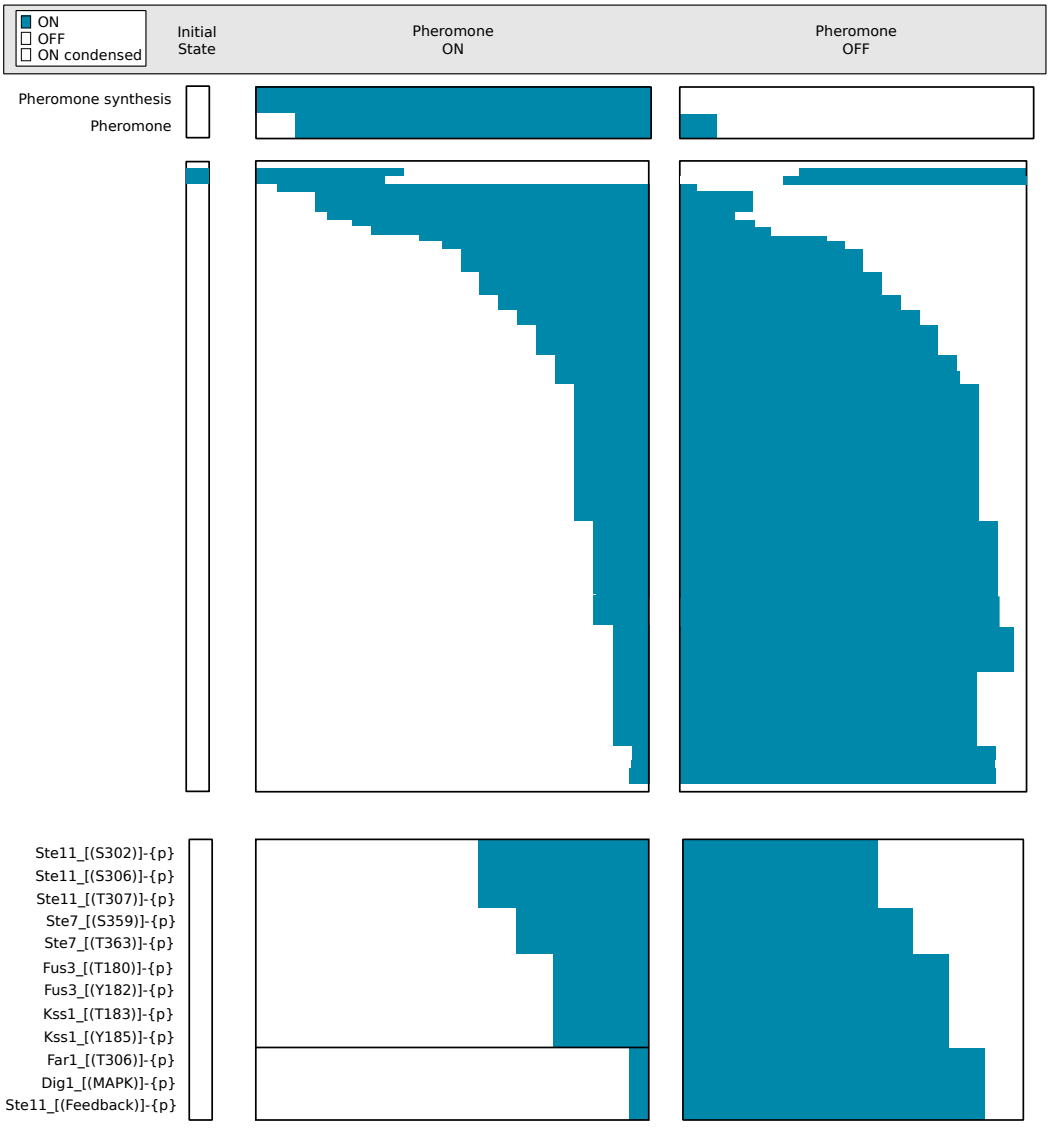


Figure 3.9: **The bipartite Boolean model simulation of pheromone response pathway.** The model is simulated using smoothing and the default parameter, until it reaches an attractor, which was used as initial state for the next simulation (Initial state). The pheromone pathway is activated afterwards by turning the pheromone synthesis ON and the model is simulated again until an attractor is reached (Pheromone ON trajectory). Next, the pheromone synthesis is turned OFF again and the model is simulated until an attractor is reached. The model responds to the input as expected.

3.10 Summary

- This chapter introduces the bipartite Boolean model for qualitative simulation and validation of large-scale mechanistically detailed signal-transduction networks.
- The logic is based on a detailed analysis of simple reaction motifs and a minimal set of assumptions.
- We define two generic update functions for reaction nodes and state nodes, which enables us to translate a rxncon network into a unique bipartite Boolean model with a defined logical table.
- The reaction layer of the rxncon reconstruction defines the update functions for state nodes, whereas the regulatory layer defines the update functions for reaction nodes.
- The generic update functions can be assembled into a complete bipartite Boolean model, predicting system level function without further parametrisation of the Boolean system.

CHAPTER 4

Rule-based modelling with rxncon

In Chapter 3 I introduced the bipartite Boolean model, which is used to validate a reconstructed network on a qualitative level by predicting the dynamics of the network on a system-level without the need for kinetic parameters, concentrations or rate laws. However, for a detailed understanding of the underlying dynamics of the reconstructed network we need quantitative models, describing the quantitative behaviour of biological molecules over time within the network. In this Chapter I describe the reinvented model generation of rule-based models that is based on rxncon 2.0 language (Chapter 2) and only loosely related to the previous rule-based model export [121]. I explain the different steps of the systematic translation of a complete defined rxncon network reconstruction into a complete defined rule-based model formulated in the BioNetGen language formalism [120].

The traditional approach to describe the concentrations of biological molecules is to enumerate over their different possible configurations as well as over the molecular reactions in a system to construct a reaction network from which we can derive a system of ordinary differential equations (ODEs) [162, 163]. Those equations give an exact description of the change in concentration of biological molecules over time. However, as discussed previously, signal-transduction networks transfer information by covalent modifications or interactions of molecular components. A single component can contain many internal molecular states, e.g. modifications at different residues, which combine combinatorially into a large number of possible configurations (microstates) [164]. Hence, the translation of a signal-transduction network into an ODE system can be difficult or even not possible in case of large networks.

A modelling formalism, describing the reconstructed network at the same level as empirical data largely avoids the combinatorial complexity problem, because experimental data describes the system on a level of macroscopic (elemental) states, e.g. only single biological structures and states (modification on a biological molecule) are measured [165].

Rule-based modelling is a powerful approach to model signal-transduction networks [120, 135, 166, 167], describing macroscopic states. The rule-based modelling formalisms encompasses two main features: 1) description of biological reactions by the 'don't write don't care' principle and 2) representation of biological reactions in terms of de-contextualized local rules [166, 168].

Those rules specify how biological molecules will change due to the application of a certain reaction. Each rule is only described by relevant structural features of the interacting molecules (reactants), e.g. binding or modification sites, while irrelevant structural features (with respect to the rule) are ignored. The definition of different rules describes the dynamics of the rule-based system [120, 169] and through the definition of only relevant features within the reactants of a rule, the rule-based approach does not enumerate over all possible microstates during the reconstruction process of the network. Hence, the rule-based approach addresses the problem of combinatorial complexity in signal-transduction networks [170, 171] that leads to a scaling network reconstruction formalism, representing the underlying empirical data in an accurate way.

The rxncon language (Chapter 2) is closely related to the rule-based language, describing the underlying data on a macroscopic (elemental) level close to empirical data. Within rxncon, we go a step further, because the rxncon language separates the reconstruction into elemental reactions and contingencies (contextual constraints on elemental reactions), which corresponds more closely to empirical data than rule-based models [121]. Moreover, rxncon is invented as a tool to develop rule-based models, leading to the fact that a rxncon network fully defines a rule-based model and that we are able to export the reconstructed model into rule-based formalisms like the BioNetGen language [121].

4.1 Rule-based modelling in general

A rule-based model can be depicted as a graph, consisting of super-nodes that represent molecules or agents, e.g. a protein. Each molecule (represented as super-node) is a collection of macromolecules (represented as nodes) of the same type, also defining the type of the molecule (molecule type). A macromolecule can contain sites (represented as sub-nodes), representing structural elements, e.g. domains or residues. Undirected edges between specific sites of molecules represent non-covalent bounds (molecular complexes). However, sites are not defined on the resolution of structural elements, i.e. no difference is made whether a site is a domain or residue. Each site has an attribute type (site type), which is a label describing the name of the structural element and an optional state property, representing internal states of a site (site state), e.g. phosphorylation.

The change of molecule properties, e.g. states within the rule-based model is defined by reaction rules. A reaction rule is a compact representation of one or more molecular reactions. Reaction rules consist of a set of molecular motifs (reactant motifs), describing reactive sites within biological molecules on the left-hand side and a set of molecular motifs (product motifs) on the right-hand side. Rules are applied on the graphs defining the molecules and maps the nodes from the left-hand side of the rule to nodes of the right-hand side of the rule to determine the outcome, e.g. an interaction [134, 172, 173]. A special feature of rules is their ability to change and re-write the system graph [174]. This is needed, because the mapping of the nodes by a rule can lead to an addition or a removal of an edge or a change of a node attribute, e.g. a state of a site. In addition, complete molecules or complexes can be added or removed through reaction rules, representing degradation or synthesis reactions. For a meaningful dynamic of the rule-based model certain rate laws are required for each rule, describing the relation between the concentration of the reactants and the rate of the reaction. Therefore, we need a formalism that enables a computer and human readable rule definition.

4.2 BioNetGen and BioNetGen Language

BioNetGen is an open source software package for constructing and simulating rule-based models [120, 175]. BioNetGen models are written in a human readable text-based format, the BioNetGen Language [175], which encodes rules in plain text. The BioNetGen Language provides a concise formalism to define models dealing with internal states, e.g. modifications as in signal-transduction networks, while largely avoiding the combinatorial complexity problem during the reconstruction process, using the 'don't write don't care' principle [176, 177]. A complete rule-based model, using the BioNetGen language formalism, requires four blocks: 1) rate constants and molecular concentrations, needed to define more complex reaction laws and to initialise the rule-base model, 2) molecular components, describing all possible sites and states of sites within a molecule, e.g. binding sites or modification states defined in the rule-based model, 3) reaction rules, defining the transition between molecules and 4) observables that are output functions relating molecules of the model to experimental data as cumulative quantities, e.g. concentration of a phosphorylated protein that are user-specified outputs of the simulation [175]. Each observable is defined by a molecular motif and a type, e.g. molecules, reflecting an ensemble of biological molecules which are typically difficult to distinguish by experiments.

A certain configuration of a molecule can only be used within rules and output functions if the configuration of the molecule is defined within its molecular component. This ensures a clean rule-based system and prevents undesired molecules within the rule-based model. Within BNGL the reaction rules are written in the same way as chemical reactions, meaning reactants on the left-hand side of the equation are transmitted into products defined on the right-hand side of the equation (Figure 4.1B). The definition of sites and states on a molecule within a rule makes the rule more stringently compared to rules with no defined sites or states. A molecule that is not stringently defined leads to a mapping on multiple different reactants which are able to satisfy the rule. This will generate multiple specific chemical reactions for each set of reactants and products, defining the conditions under which the respective molecular reactions takes place. In a fully defined rule-based model all rules are parametrised by respective rate constants, defined in the block for parameter and concentrations.

4.3 Translation of a rxncon reconstruction into a rule-based model

We implemented a systematic procedure to generate rule-based models in the BioNetGen Language formalism, enabling the export of reconstructed rxncon models into a rule-based model (Algorithm 1).

In this section I exemplify the translation process on a rxncon reconstruction of the insulin pathway. To demonstrate different model behaviour we reconstructed two different versions of the insulin pathway. One version without dephosphorylation reactions and one with dephosphorylation reactions. However, rule-based models need to be parametrised to reproduce the dynamics of the underlying mechanism of a reconstructed network in a meaningful way, which is not trivial and falls outside of the scope of this work. Note, that all BioNetGen files are exported according to the conventions of BioNetGen 2.2.6. In the following text, I use the terms rule-based model and BioNetGen model interchangeably. For the construction of a valid rule-based model we need the following information: *molecule types*, *reaction rules*, *parameters*,

Algorithm 1 rule-based model from rxncon

```
1: mol_defs = the molecule types within the rxncon system
2: function rule_based_model_from_rxncon(rxncon_system)
3:     rules = []
4:     for reaction in rxncon_system if not output reaction do
5:         quan_cons = intersection of all quantitative contingencies of reaction
6:         qual_cons = intersection of all qualitative contingencies of reaction
7:         for quan_con in quan_cons do
8:             con_set = intersection of qual_cons and quan_con
9:             con_set = add molecule connectivity information as additional contingencies
10:            solutions = calculate the valid solutions of con_set
11:            solutions = remove all global states from solutions
12:            for solution in solutions do
13:                positive_solutions = calculate the complement of negated elemental states
   within solution
14:                positive_solutions = remove solution with mutually exclusive elemental states
15:            end for
16:            for positive_solution in positive_solutions do
17:                rule = calculate the rules from reaction and positive_solution
18:                if rule is not equivalent to a known rule in rules then
19:                    rules = append the rule
20:                end if
21:            end for
22:        end for
23:    end for
24:    initial_conditions = calculate the default initial conditions
25:    observables = calculate the observables
26:    return RuleBasedModel(mol_defs, inital_conditions, observables, rules)
27: end function
```

seeded species, *observables*. In the next subsections I explain the different categories and how we generate them from a rxncon system.

4.3.1 Molecule Types

To calculate the molecule types of a rule-based system we need to know: 1) which rxncon components are in the system and 2) what kind of internal states do they have. The rxncon language describes biological processes by de-contextualized elemental reactions and contextual constraints (contingencies) [121, 124]. An elemental reaction consists of rxncon specifications (reactants), which are basic elements of the rxncon language, representing biological molecules and therefore, correspond to molecules in a rule-based model. More precisely, a rxncon specification defined on a component-level (rxncon component) describes the type and the name of the rule-based molecule. In addition, an elemental reaction produces, consumes, synthesises or degrade one or more elemental state, containing the information to determine the sites and the state of the site of a rule-based molecule (Figure 4.1).

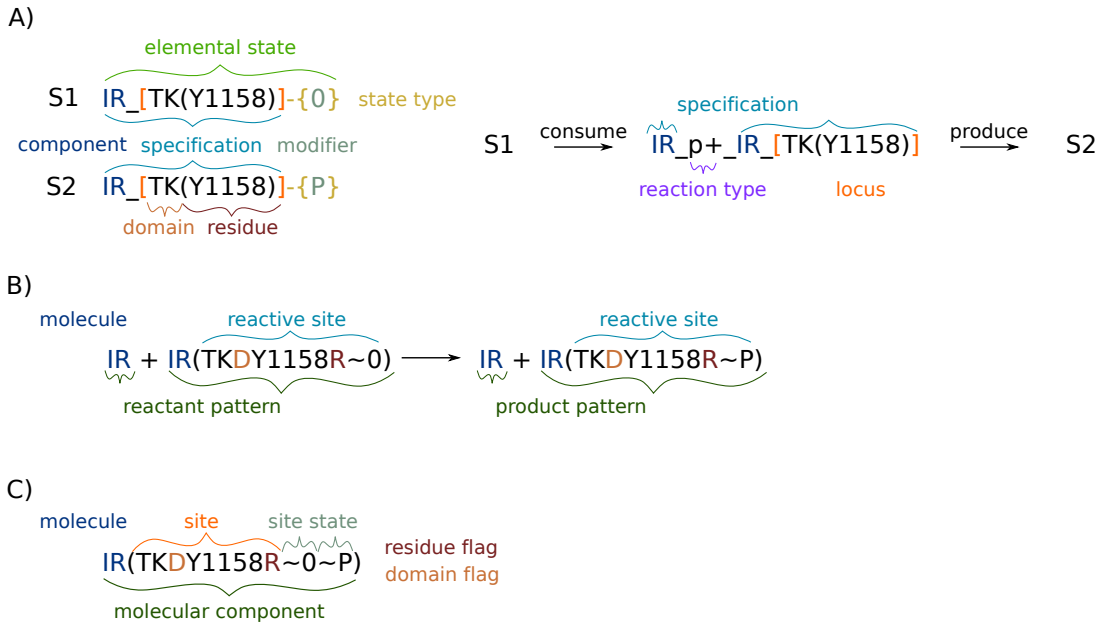


Figure 4.1: **Analogy between rxncon and BNGL** A) Exemplified syntax of rxncon states and rxncon reactions. A rxncon modification reaction consumes a neutral elemental state (S1) and produces a modified elemental state (S2). B) Exemplified syntax of a rule of the same reaction as shown in panel A. C) Exemplified molecule component. The two states ∼0∼p of the molecule site TKDY1158R indicates that the site can be unmodified or phosphorylated.

The information given by the rxncon reactions is sufficient to define the molecular component. If we iterate over the complete list of elemental reactions within the rxncon system, we can retrieve all rxncon components within the rxncon system (Algorithm 1 line 1), e.g. IR, and which elemental states belong to a rxncon component, e.g. IR_[TK(Y1158)]-{p}. The locus information of the internal rxncon states is described in their rxncon specification and can be

used to group rxncon components by, e.g. residue or domain name. The locus information is used to determine the sites of a rule-based molecule, whereas the rxncon state type, e.g. phosphorylation state, is used to define the states of the different molecule sites. For instance, the reaction `IR_p+_IR_[TK(Y1158)]`, consumes the elemental state `IR_[TK(Y1158)]-{0}` and produces the elemental state `IR_[TK(Y1158)]-{p}`. These two rxncon states describe two different properties of the same residue (`Y1158`), which will be translated to different internal states on a single molecule site of the `IR` molecule.

Note, that within BNGL we do not distinguish between different molecule resolutions like domains or residues. If they have the same name both will be handled as the same site. However, rxncon distinguishes between the different resolutions (see Chapter 2). To avoid a clash of names we append `D` or `R` for domain or residue, respectively during the translation step of a rxncon system into a complete rule-based system. For instance, the site `TKD` corresponds to the reconstructed domain `TK` and the site `TKDY1158R` corresponds to the reconstructed residue `Y1158` at domain `TK` (Figure 4.1C).

4.3.2 Reaction rules

The change of an elemental state property and therefore, the type of the reaction is defined by the skeleton rule. Hence, an elemental reaction is defined by the elemental states that are changing between the left-hand side and the right-hand side of the skeleton rule. A single elemental reaction (together with its skeleton rule) corresponds to the reaction centre of a fully de-contextualized local rule of a rule-based model [175], encompassing the molecules that change during the evaluation of the reaction rule [178].

The contextual constraints on a rxncon reaction are given by its contingencies and therefore, correspond to the reaction context of a rule within a rule-based model, describing rule-based molecules that affect the reaction rule but remain unchanged [178]. An elemental reaction, combined with its contingencies, can be translated into one or more rules (Algorithm 1 line 2-23). For instance, a rxncon reaction with no contingency or only a single qualitative contingency with a required rxncon state is translated in one reaction centre with one reaction context and therefore, in one rule. We have to give special attention on explicit or implicit Boolean contingency OR statements, generating a number of different rules. For instance, if the resolution of an elemental state appearing in a contingency term, is not elemental:

```
A_p+_B_[(r)]; !  A-{p}
C_p+_A_[(x)]
C_p+_A_[(y)]
```

the contingency statement will be translated into a Boolean term OR of elemental states, meaning that the rxncon reaction (`A_p+_B_[(r)]`) requires a phosphorylation at residue `x` or residue `y`. This results in two rules:

```
A(xR~p) + B(rR~0) -> A(xR~p) + B(rR~p)
A(xR~0, yR~p) + B(rR~0) -> A(xR~0, yR~p) + B(rR~p)
```

The first rule derived from the rxncon reaction `A_p+_B_[(r)]` will have a molecule `A` with the state `xR~p` (phosphorylation at site `xR`). The second rule will have a molecule `A` with the state `yR~p` (site `yR` is phosphorylated) and should state explicitly that site `xR` is not phospho-

rylated (xR∼0) to avoid overlaps between the rules and therefore, double considerations of the same molecule in the rule-based model. In case of an inhibition by an elemental state or an elemental state in context of a Boolean statement NOT, the complement of the elemental state will be used and all the complements will be connected by the Boolean statement OR. For instance, if we have an elemental state which can have two different modifications on the same residue, e.g. a phosphorylation or ubiquitination, we will have three different forms of this elemental state A_[(r)]-{p}, A_[(r)]-{ub}, A_[(r)]-{0}. The complement of one of them results in a Boolean statement OR of the other two.

Quantitative contingencies (Chapter 2) imply that the reaction rate can be increased or decreased by the respective contingency Effector, which in turn describes an implicit rxncon Boolean term OR, creating two different rules. One rule requires the contingency Effector, e.g. phosphorylation site, enabling an increase or decrease of the reaction rate, whereas the other rule explicitly excludes the contingency Effector, e.g. unmodified site, representing the enhanced or suppressed reaction context (Algorithm 1 line 7-22). This leads to two disjunct rules, giving the modeller the possibility to change the respective reaction rates according to the biological meaning.

However, if we combine more than one quantitative contingency term, we have to make sure that we do not consider the same molecule in the rule-based model twice. Therefore, we use picoSAT [142] to build the solutions of the Boolean contingency terms, ensuring a disjunctive reaction context of all rules derived from an elemental reaction and its contingencies. However, using picoSAT for the disjunction process can lead to non-connected components or mutually exclusive states within a solution. This can be avoided by adding additional contingencies to the reaction to make sure that all rxncon components of the rxncon states within the contingencies are directly or indirectly connected to the reactants of the rxncon reaction and to discard solutions, containing rxncon states which are mutually exclusive to each other. Hence, all rules generated automatically from the rxncon system have a disjunct reaction context and the molecules within a rule are connected to the reactants and not mutually exclusive on a single-molecule level.

Out of simplicity reasons we assume mass-action kinetics for all the generated rules. Mass-action kinetics define the rate of a reaction being proportional to the product of the reactant concentration [179]. This is derived from the collision theory under the assumption of well-mixed conditions and that the reaction occurs in one step, e.g the molecular reaction:

ProteinA + ProteinB $\longrightarrow$ ProteinC

has the rate:

k1[ProteinA][ProteinB]

where k1 is a rate constant and [ProteinA], [ProteinB] are the concentrations of ProteinA and ProteinB, respectively. This reaction kinetic is used in systems where rates are interpreted as a continues flux of mass through the reaction, e.g. in systems with large particle concentrations. However, it can also be applied to discrete systems, where the rate can be interpreted as a probability that the reaction can occur in a respective time step [180].

4.3.3 Parameters

The parameter section within a BioNetGen file describes the rate constants of a rule as well as the initial amounts of the different molecules within the rule-based system. We decided to create a generalised parameter for all rules, holding the value for its rate constant. The generalised parameter is called `k` and has a default value of 1.0. For bidirectional reactions, e.g. protein-protein-interactions, we split the reaction according to the BNGL conventions into two unidirectional reactions, one for the association and one for the dissociation reaction rule, and assign a reaction rate constant to each. In addition, we add a parameter for each molecule holding the initial particle count of the respective molecules, e.g. `NumIR` 1000, which sets the default value of the molecule count (`Num`) of the insulin receptor (`IR`) to the default value of 1000 (Algorithm 1 line 24). However, the modeller is free to change these parameters.

4.3.4 Seeded species

The simulation of a rule-based model uses the molecules within the seeded species section. We decided to define the starting point of each simulation with the fully neutral form of the defined molecule types, meaning that we initialise the simulation with the unmodified (indicated by ∼0), unbound form of the defined molecules (Algorithm 1 line 24). Each elemental state, describing a modified or bound biological molecule, has a neutral state as counterpart (Chapter 2). To create the molecules within the seeded species section we collect the neutral information of each molecule described in the rxncon system and create the fully neutral form. For instance, for the insulin receptor (`IR`) we can identify the `lig`, the `JM` and the `IRBD` binding domains as well as the residues `Y1158`, `Y1162`, `Y1163` and `Y972`, where the first residues are defined within the `TK` domain and the last residue is defined within the `JM` binding domain. Hence, the insulin receptor is initialised as:

```
IR(IRBDD,JMD,JMDY972R∼0,TKDY1158R∼0,TKDY1162R∼0,TKDY1163R∼0,ligD)
```

The seeded molecules of the rule-base model are initialised with a certain number of particles of the respective neutral form. The number is defined in the parameter section (default 1000). Since the model should show the influence of insulin to the system, the particle number of unbound insulin is initialised with 0.

4.3.5 Observables

Elemental states are macroscopic states corresponding to independent observable quantities, describing a molecular state property of one biological molecule, e.g. phosphorylated protein. To observe molecules with certain properties within a rule-based, model output reactions can be defined in the rxncon model. Output reactions are only defined within the contingency list. Elemental states, which are required for a certain output reaction, are translated into observables of the rule-based system (Algorithm 1 line 25). The observables or output signals can be defined within the Observables section. They are sampled during the entire simulation and represented as cumulative quantities, giving the opportunity to relate molecules of the model to experimental measurements.

Within the reconstructed insulin model we defined output reactions for activation of the PI3K and the Ras pathway. The PI3K signal is activated by the binding of IRS and PI3K [66]:

```
[PI3K]; !  IRS_[bd]--PI3K_[SH2]
```

and described by the observable:

```
Molecules PI3K0 IRS(bdD!1).PI3K(SH2D!1)
```

The observable is defined as a complex (indicated by '.'), consisting of two molecules `IRS` and `PI3K`, which are connected by a bond name (indicated by '!' followed by a number, e.g. *1*), indicating that both molecules bind to each other through the sites `bdD` and `SH2D`.

The Ras pathway on the other hand is activated by a more complicated mechanism. It only becomes active if Grb2 is bound to SOS and Grb2 is bound to either Shc or IRS [66]. This results in the contingency term:

```
[RAS]; !  <Grb2-SOS>
<Grb2-SOS>; AND Grb2--SOS; AND <GS>
<GS>; OR Grb2_[SH2]--Shc_[bd2]; OR Grb2_[SH2]--IRS_[bd]
```

BNGL does not allow for algebraic equations. Hence, we append numbers to the resulting observables. In this case, a `RAS0` and a `RAS1` observable will be created:

```
Molecules RAS0 Grb2(SH2D!1,SOSD!2).SOS(Grb2D!2).Shc(bd2D!1)
Molecules RAS1 Grb2(SH2D!1,SOSD!2).IRS(bdD!1).SOS(Grb2D!2)
```

To get the total activity of the Ras pathway, the result of both observables has to be added. For the simulation of the insulin pathway we want to track the insulin concentration during the simulation but it is not an output of our model. Hence, we included insulin manually:

```
Molecules INSULIN insulin
```

The information collected above results in a rule-based model (Algorithm 1 line 26).

4.4 ODE simulation with BioNetGen

The rules of a rule-based system describe the reconstructed molecular network and can be used to simulate the network dynamics directly via an ODE system.

The rxncon reconstruction of the insulin pathway without dephosphorylations consist of 16 rxncon reactions and 20 contingencies (Supplementary File SF8). This model was translated into a rule-based model with eight components, 41 rules and three observables (Supplementary File SF9). Subsequently, BioNetGen was used to perform an ODE simulation of the rule-based model. The ODE system contains 2597 molecules which are connected by 30839 reactions. The system was simulated for 150 time units in total. We started the simulation without insulin for 50 time units, then added 1000 molecules of the neutral form of insulin and run the simulation for another 50 time units. Subsequently, insulin was removed from the system and we run the simulation for 50 time units (Figure 4.2).

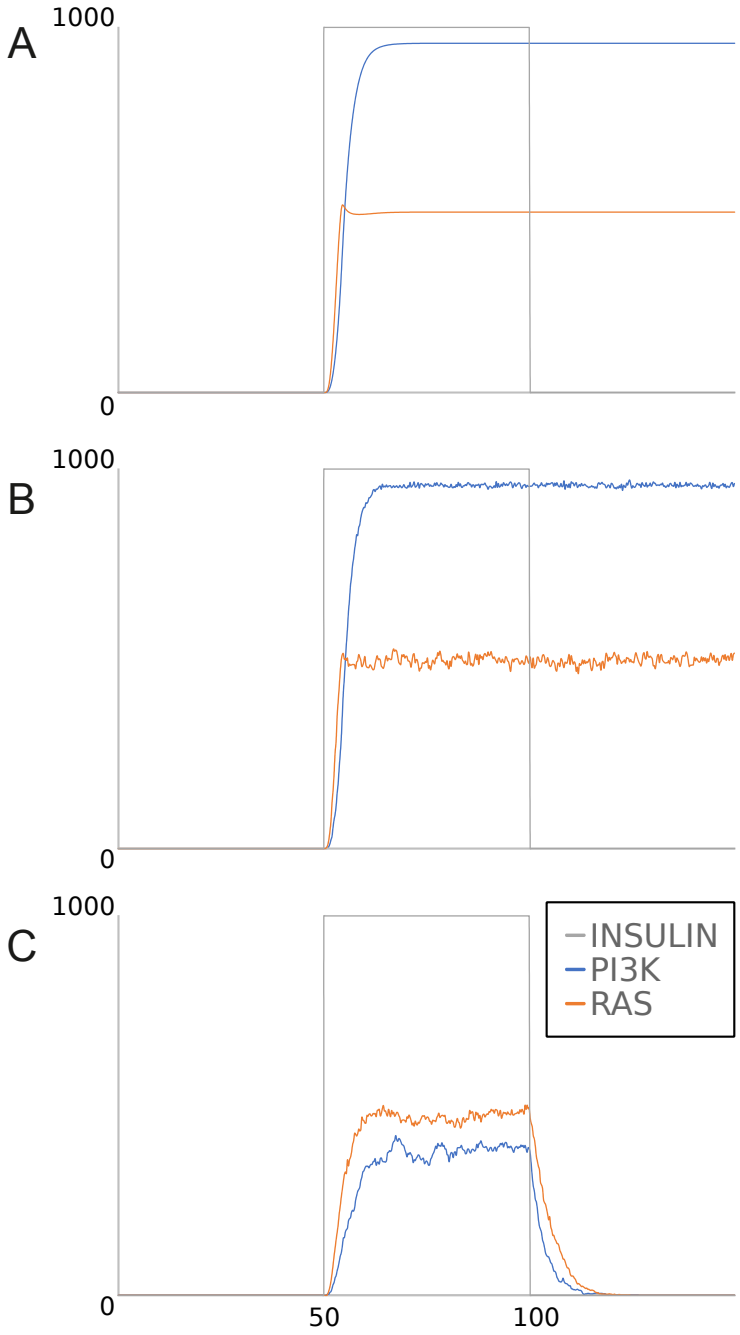


Figure 4.2: **Simulation of the exported insulin rule-based model.**A) ODE simulation of the insulin rule-based model in the absence of dephosphorylation reactions. After insulin was given to the system the molecule number of both output signals (PI3K and RAS) increases. After removal of insulin signal both output signals do not respond to the change. B) NFsim simulation of the insulin rule-based model in the absence of dephosphorylation reactions. The behaviour is similar to the ODE solution in panel A, except for the stochastic noise. C) NFsim simulation of the insulin rule-based model with seven additional dephosphorylation reactions. The particle number of both output signals (PI3K and RAS) increases with an increase in insulin and decrease if insulin is removed from the system. The molecule number is lower compared to the simulation in panel B, because the output signals require fully phosphorylated complexes which decrease in the presence of unregulated dephosphorylation reactions. To get a decent signal to noise ratio within the NFsim simulation, the molecule counts of all phosphatases were set to 10 and the rate constants for the dephosphorylation reactions were set to 0.025.

We added an additional rule to the model to completely remove insulin from the rule-based system:

```
Insulin -> 0 ins_deg_0 DeleteMolecules
```

The DeletedMolecules keyword breaks a complex into its subunits by removing the target molecules, in our case any form of the insulin molecule. This ensures that the insulin only gets degraded if the degraded insulin molecule is connected to another molecule, e.g. `IR` of the rule-based system and the other molecule is released back to the rule-based system. In addition, we added the parameter `ins_deg_0` with an initial value of 0. As shown in Figure 4.2A the molecule number of the output signals (Ras and PI3K) does not decrease after removing insulin from the rule-based system. That behaviour is expected, because the output signal requires fully phosphorylated complexes and we did not define any reaction removing those phosphorylations, which coincides with the Boolean simulation of this pathway (Supplementary Figure S1, Supplementary File SF10).

Next, we simulated the reconstructed insulin network including 7 additional dephosphorylation reactions (Supplementary File SF2, SF11), using the initial conditions and BNGL commands as for the first simulation. However, after 12 iterations of the network building process the reaction network consisted of 125172 molecules connected by 492015 reactions. Hence, we stopped the process after 12 iterations, because the network got too big and it was unknown when the network generation process would terminate.

For larger networks BNGL also supports an interface for network-free simulations with NFsim [181] (further reading [182]).

4.5 NFsim and network free simulation

A rule-based model is a compact representation of a reconstructed biological network. However, for simulation purpose the rules have to be enumerated and expanded to a complete network of the reconstructed system to map it onto ODEs, which might lead to a combinatorial problem. Hence, it is often impossible to simulate a fully defined signal-transduction network, using an ODE simulation for a rule-based model, leading to the requirement of alternative approaches. BioNetGen supports an interface for the network-free simulation tool NFsim [181], which is useful for large signal-transduction networks. A network-free simulator avoids the generation of a complete network defined by the rules before the actual simulation begins [183, 184]. This requires the change of the system state perspective from a population (concentration) perspective to a particle-based perspective (molecules with reactive sites). The basic idea is that it is more efficient to work on the level of particles than to enumerate all the distinct microstates, because the number of particles is less than the total number of fully defined molecules.

Network-free methods are purely stochastic, meaning that the simulation results are not deterministic. However, all currently available methods are similar [185] and the following steps can be found in different network-free algorithms: 1) calculation of rates (propensities) of the different rules by counting the number of reactive sites matching the molecular motifs on the left-hand side of the rule, 2) at each step of the simulation, rules are selected randomly based on the relative rates calculated in the first step and 3) the reactant set is randomly selected following a uniform distribution among the possible sets of reactants and transformed in accordance to the rule. Note, we used the NFsim version 1.11 for all network-free simulations.

4.6 Network-free simulation with NFsim

The NFsim simulator was used to change the simulation method from ODE to network-free simulation. Previous to the simulation of the reconstructed insulin network with dephosphorylations I show that the ODE simulation and the network-free simulation are predicting the same behaviour of the input (insulin) and output signals (PI3K, Ras).

To add a number of unbound insulin molecules (initialised with 0), and to remove them at a later time point within a network-free simulation, we extended the rule-based model by adding a synthesis reaction, synthesising insulin in its neutral from and a degradation reaction, degrading any insulin:

```
0 -> insulin(IRD) insulin_prod()
insulin -> 0 insulin_deg() DeleteMolecules
```

In addition, both rules are regulated by the parameters `insulin_prod()` and `insulin_deg()`, which are modelled as functions:

```
insulin_prod() = ins_prod_0 * abs(Numinsulin - INSULIN)
insulin_deg() = ins_deg_0 * INSULIN
```

The functions `insulin_prod()` and `insulin_deg()` are defined within an optional functions block within the BNGL rule-based model. Within these functions one variable and three parameters are defined: the observable `INSULIN` (as defined above) and the parameters `Numinsulin`, `ins_prod_0` and `ins_deg_0`. `Numinsulin` is the maximum number of insulin molecules within the system (default 1000). Both rate constants, `ins_prod_0` and `ins_deg_0`, are initialised with 0 within the parameter section. The functional form of the reaction rate allows to control the synthesis and degradation of insulin by controlling the respective rate constants within the function. If we set the rate constant `ins_prod_0` to a much higher number, compared to the other rate constant `ins_deg_0`, the synthesis will be active when the number of insulin molecules (any form of insulin) is smaller than the maximum number insulin molecules (`Numinsulin`). If the number of insulin molecules is the same as the maximum number of insulin molecules, the synthesis reaction will turn OFF. The degradation reaction does not depend on the maximum number of insulin molecules in the system, meaning that if we set the rate constant `ins_prod_0` to a much higher number compared to `ins_prod_0`, insulin will be degraded as long as there are insulin molecules left in the rule-based system.

The rule-based system is simulated using a Run NF script (Supplementary File SF12). The system is simulated for 50 time units without insulin. Afterwards the synthesis rate constant `ins_prod_0` is increased to 100 (the rate constant for the degradation is unchanged) and simulated for another 50 time units, followed by the decrease of the synthesis rate constant to 0 and an increase of the degradation rate constant `ins_deg_0` to 100, which is simulated until the simulation time of 150 time units is reached. The behaviour of the ODE simulation and the network-free simulation are similar except of some statistical noise (Figure 4.2B). Hence, we run the extended model with the added seven dephosphorylation reactions with the same procedure as described above. The output signals (PI3K and Ras) respond to insulin as shown in the bipartite Boolean model (Supplementary Figure S1). The particle number of both output signals (PI3K and RAS) increases with an increase in insulin and decrease if insulin is removed from the system (Figure 4.2C). Hence, we are able to export a reconstructed rxncon model into

a valid and functional rule-based model.

4.7 Summary

- This chapter introduces the new developed systematised process to translate a rxncon system into a quantitative rule-based model in BNGL formalism.
- A rxncon reaction describes the reaction centre of one or more rules.
- A contingency describes the reaction context of one or more rules.
- The unambiguous interpretation of the rxncon syntax, enables us to translate a rxncon statement into one or more non-overlapping rule(s) of a rule-based model.
- Adding new information to the rxncon model and exporting it into a rule-based model in BNGL format, enables an iterative building process of rule-based models.

CHAPTER 5

Discussion and Outlook

In this thesis, I presented a five step workflow to develop quantitative models. In step one and two of this workflow, the scope of the network is defined and a first seeded version of the rxncon model is created. The seeded model can be refined later by adding more mechanistic information. In step three and four, the rxncon model is translated into a ready to simulate qualitative bipartite Boolean model, an approach that allows parameter free structural validation and simulation of large-scale signal transduction networks. In the last step of this workflow, the rxncon model is translated into a rule-based model enabling quantitative simulations. This workflow was described in 3 Chapters each with its own perspective within the iterative reconstruction process.

Reconstruction step

Since the first release of rxncon [121], we worked on the generalisation of the language and improved the model generation logic by introducing two new concepts within the rxncon language [124]: the resolution of specifications that enables a more precise definition of the mutual exclusivity and the concept of skeleton rules, describing pattern of effects on reactants that enables users to define their own rxncon reactions during the reconstruction process. In addition, we adapted the input format and changed it into SBtab [186], a promising standard for table definition of models and data in the field of systems biology to facilitate annotation and exchange of the model. Through the introduction of those concepts and the adaptation of the input format we could improve the model generation logic and increased the customisation of the language.

We redefined the meaning of contingencies on bidirectional rxncon reactions: if a lumped forward and reverse rxncon reaction is used, e.g. protein-protein interactions, by removing the directionality sign (+/-), the contingency has to be assigned explicitly to the reverse rxncon reaction to have an influence on the reverse rxncon reaction. Otherwise we assume that the regulation is only set for the forward rxncon reaction and the reverse rxncon reaction is handled

as unregulated. This split allows a definition for the influence of contingencies on rxncon reactions by keeping the feature of lumping bidirectional rxncon reactions of the old language.

Contingencies can now be defined on a less specific rxncon state resolution, which may map to several rxncon states, depending on the available information. This can be useful to capture literature data at different resolutions or to simplify contingencies but has no influence on the fact that a rxncon state can be defined precise and unique if needed. Through the introduction of complex structuring, we enable the user to add an accurate description of biological complexes and reactions, e.g. allow the definition of cis and trans modifications and interactions.

The rxncon language describes the reconstructed network on a molecular level. Hence, it is difficult to model higher level mechanisms, e.g. vesicle transport or the change of global quantities like pH influencing the system. To overcome those limitations the reconstructions can be adapted by using global input states and output reactions, to describe complex mechanisms as it was done for our cyclic Hog example. However, the definition of global input states does not fit the concept of molecules within rule-based models. Alternatively, new rxncon reactions can be defined by the user during the reconstruction process to describe macromolecular processes not yet captured. Hence, through the flexibility of rxncon we have the potential to precisely define new events within the rxncon language and therefore, reconstruct fully functional and detailed defined large-scale signal transduction networks, which will help to improve the mechanistic understanding of processes within signal transduction networks.

Qualitative validation and simulation step

The bipartite Boolean model was developed for qualitative validation and simulation of large-scale mechanistically detailed signal-transduction networks. The formalism is based on the Boolean logic and can be exported into a simulatable model, e.g. with the BoolNet package. The new logic is based on a detailed analysis of simple reaction motifs and a minimal set of assumptions. By defining two generic update functions for reaction nodes and state nodes we are able to translate a rxncon network into a unique bipartite Boolean model while the regulatory structure of the rxncon reconstruction enables us to define a logical table. The reaction layer of the rxncon reconstruction defines the update functions for state nodes, while the regulatory layer defines the update functions for reaction nodes. These different building blocks can be assembled into a complete bipartite Boolean model, predicting system-level function without further parametrisation of the Boolean system.

The functionality of signal-transduction pathways often does not need parametrisation in terms of rate laws, reaction constants and relative concentrations. Hence, a qualitative representation of the regulatory layer is often sufficient. The bipartite Boolean model can be used to predict the qualitative dynamic behaviour of a biological system and therefore, provides a tool to validate the model structure without prior parametrisation. If the minimum requirements (input-output relation) are not fulfilled, the network lacks mechanistic information which should be added during the reconstruction process. The reconstruction and validation steps can be performed iteratively until the Boolean model behaves according to the experimental *in vivo* results. Hence, it can act as a control step before building rule-based models and start the process of parametrisation.

The bipartite Boolean model formalism is a complete reinvention of the previously developed bBM [144]. The new formalism is based on the second generation rxncon language [124]. The

rxncon reactions semantic is encoded in the skeleton rules, which are important to map rxncon reactions to the generic update functions of the bipartite Boolean model. Moreover, the new bipartite Boolean model explicitly includes neutral state nodes, representing neutral elemental states, such as unmodified or unbound state nodes, which enables mutually exclusive state nodes at the single-molecule level to be present simultaneously at the system-level. Some of these neutral state nodes might appear unregulated in analysed models due to: 1) a constant pool of unmodified biological components in the cell, e.g. if there is a constant turnover like a synthesis reaction in the biological system, 2) an experimental bias, e.g. in yeast we know much more about the modifying reactions than about the reactions that reverse the modification (e.g. phosphorylation, de-phosphorylation) [121]. In addition to changes within the rxncon language, we engineered the update functions of the new bipartite Boolean model from scratch, leading to three important improvements: 1) the update functions for state nodes mirror the reaction layer, including the dependence on source state nodes for reaction nodes, 2) the synthesis and degradation of rxncon components map on the respective elemental state nodes and 3) Boolean components within a Boolean system are not independent model entities.

In spite of the more detailed description of the new bipartite Boolean logic it was shown that the original simulation method worked in many cases [136, 144, 187]. This indicates that signal-transduction pathways are rather robust. However, the previous bBM had some issues limiting the modelling formalism. For example, there was a bias towards the modified state nodes, in the previous version. Only modified elemental states were considered explicitly and neutral states kept implicit. Furthermore, each elemental state had exactly one unique complement, hence, it was not possible to model a Boolean system, where modified and neutral state nodes can coexist even at the system level. Elemental states were not intrinsically mutually exclusive, resulting in cases where alternative modification and interaction partners had to be indicated with additional inhibitory contingencies during the reconstruction process, making the reconstruction unnecessary complicated and error prone. Additionally, the degradation of state nodes was not considered in detail in the previous version, e.g. if a rxncon component was degraded the state nodes and reaction nodes depending on the rxncon component remained active. Hence, the original rxncon language and bBM logic were a useful approximation, describing the key features of the Boolean system but the rxncon 2.0 language and the new bipartite Boolean model logic are syntactically and semantically more accurate.

A bipartite Boolean model cannot reproduce the qualitative outcome of every dynamical system. Problems arise if a biological reaction on a single molecule has a much smaller rate than all the other biological reactions, because in this case we are not able to have two active mutually exclusive state nodes at the same time. However, in those cases we could interpret the biological molecule as not abundant enough to be functionally relevant and the respective biological reaction can be considered as functionally inactive. Beside, the bipartite Boolean model assumption of rate equilibrium results in dominance of some reaction nodes, limiting the expressiveness of the formalism, because it is not possible to simulate a cyclic system without any inhibiting contingencies. Hence, the interplay between the different qualitative contingency types (x/!) represent the only degree of freedom which can be used for parametrisation. This option did not exists before and was also not necessary, because elemental states and their complements could not coexist in the previous version. The unambiguous interpretation of the rxncon syntax, enables the translation of large-scale signal transduction networks, reconstructed based on the reaction-contingency formalism, into a fully functional bipartite Boolean model.

Quantitative modelling step

The reconstructed network can be translated into a quantitative rule-based model described in the BNGL formalism. Most approaches modelling signal-transduction networks require the user to define a microstate level of a molecule which leads to limitations of those approaches by combinatorial complexity. Rule-based models are able to generate a complete network of biological reactions, based on a number of user defined rules. The rxncon language is closely related to the rule-based modelling approach, because a rule-based model as well as the rxncon language avoids the combinatorial complexity by 1) describing the biochemical reactions through de-contextualized rules and 2) specifying only reaction relevant molecule information. We are able to translate user defined elemental reactions and contingencies automatically into rules. This is possible, because a rxncon reaction describes the reaction centre of a rule, whereas a contingency describes the reaction context. The interpretation of the rxncon syntax, enables us to translate a rxncon statement into one or more non-overlapping rule(s) of a rule-based model, which in turn allows an iterative building of rule-based models by adding new information to the rxncon model and exporting it into a rule-based model in BNGL format. Hence, rxncon provides a way to build rule-based models in a systematic way based on any complete and valid defined rxncon system.

The BNGL formalism does not distinguish between the resolution of different molecular structures, e.g. domains and residues, limiting the expressiveness of the model, because domains and residues with the same name are corresponding to the same sites. Additionally, the BNGL formalism does not allow algebraic expression in the observable section, making the observation of the same outcome under different conditions, e.g. the activation of a signal cascade by different proteins, difficult. The BioNetGen software package allows the simulation of the rule-based model as an ODE system. Therefore, all possible molecular configurations of the reactants defined in the rule-based system are enumerated to simulate the network dynamics. In other words, it first calculates the entire molecular state space before it is able to simulate the dynamics of the system, leading to the same issues other approaches have - the combinatorial problem. However, there are possibilities to avoid an ODE based simulation, because BioNetGen provides an interface for network-free simulations, which is not based on a complete reaction network, but able to calculate the needed parts of the network during runtime, enabling the simulation of mechanistic detailed and large scaling networks.

Some functionalities which can be found in rule-based models cannot be covered by the rxncon language. For instance, rule-based models require quantitative information to meaningful predict biological behaviour, which cannot be defined within the rxncon language. Therefore, we provide default values for rate constants and initial concentrations during the export process but those values are arbitrary and have to be changed manually in the BNGL file. In addition to that, it is not possible to express processes by functions as it is possible within the BNGL formalism, e.g. reaction rates in form of functions, allowing a dynamic control over rules.

Outlook

Several concepts, e.g. localisation and allele effect, which are crucial for signalling processes are missing and will be subject of further studies. The implementation of additional mathematical modelling formalisms (kappa) as well as graphical visualisation formalisms (Systems Biology Graphical Notation) will also improve rxncon.

rxncon is currently only accessible by a python library or via scripts, which can be executed on the command line. Hence, the current implementation requires the user to have a basic understanding about programming and the command line on its respective operation systems. Therefore, we developed a more user-friendly environment based on a graphical interface, enabling the iterative reconstruction of signal-transduction networks outside the command line and make rxncon accessible to a more general scientific community. This workbench has to be tested intensively and should be adapted to the needs of the user as well as to further developments in rxncon.

Working with rxncon has several advantages compared to other network reconstruction methods like the BNGL formalism: 1) rxncon is closer related to empirical data than rule-based models, leading to a more precise description of the underlying mechanistic of the reconstructed network, 2) rxncon annotates a knowledge database that is human and computer readable, enabling the reuse and merge of reconstructed models [188], 3) rxncon enables the export into different mathematical models, e.g. bipartite Boolean or rule-based models and into different compact visualisation formats, enabling a mathematical and visual simulation and validation of the reconstructed network and 4) rxncon enables the iterative building of rule-based models.

Taken together, we developed a language for reconstructing large-scale biological networks. The rxncon language largely avoids the combinatorial complexity problem, which can occur during the reconstruction of signal-transduction networks. This is achieved due to the context-free grammar within the rxncon language as well as through the description of the data on the same level as biological findings. With the development of rxncon 2.0, we improved the expressiveness and precision of the rxncon language furthermore, which is now on the same level as a rule-based modelling language. Additionally, we implemented a python library that acts as an interpretation engine for the rxncon language. It enables the compilation of a rxncon 2.0 network definition into a bipartite Boolean model (further reading [189]) as well as into a rule based model in the BNGL language [122]. Furthermore, reconstructed rxncon networks can be exported into different visualisation formats. Hence, rxncon is able to bridge large-scale network reconstruction and classical mathematical modelling approaches.

Bibliography

[1] M. D. Mesarović. Systems theory and biologyâĂŤview of a theoretician. In *Systems theory and biology*, pages 59–87. Springer, 1968.

[2] M. Mesarovic, S. Sreenath, and J. Keene. Search for organising principles: understanding in systems biology. *Systems biology*, 1(1):19–27, 2004.

[3] R. Stichweh. Systems theory. *B. Badie, D. Berg-Schlosser & L. Morlino. International encyclopedia of political science*, 8:2579–2582, 2011.

[4] H. Kitano. Computational systems biology. *Nature*, 420(6912):206–210, 2002.

[5] T. Ideker, T. Galitski, and L. Hood. A new approach to decoding life: systems biology. *Annual review of genomics and human genetics*, 2(1):343–372, 2001.

[6] S. Kesić. Systems biology, emergence and antireductionism. *Saudi journal of biological sciences*, 23(5):584–591, 2016.

[7] H. V. Westerhoff and B. O. Palsson. The evolution of molecular biology into systems biology. *Nature biotechnology*, 22(10):1249–1252, 2004.

[8] M. A. Hibbs, C. L. Myers, C. Huttenhower, D. C. Hess, K. Li, A. A. Caudy, and O. G. Troyanskaya. Directing experimental biology: a case study in mitochondrial biogenesis. *PLoS Comput Biol*, 5(3):e1000322, 2009.

[9] J. D. Orth and B. Ø. Palsson. Systematizing the generation of missing metabolic knowledge. *Biotechnology and bioengineering*, 107(3):403–412, 2010.

[10] A. Mogilner, J. Allard, and R. Wollman. Cell polarity: quantitative modeling as a tool in cell biology. *Science*, 336(6078):175–179, 2012.

[11] J. R. Karr, J. C. Sanghvi, D. N. Macklin, M. V. Gutschow, J. M. Jacobs, B. Bolival, N. Assad-Garcia, J. I. Glass, and M. W. Covert. A whole-cell computational model predicts phenotype from genotype. *Cell*, 150(2):389–401, 2012.

[12] C. Kreutz and J. Timmer. Systems biology: experimental design. *FEBS journal*, 276(4):923–942, 2009.

[13] E. J. OâĂŹBrien, J. M. Monk, and B. O. Palsson. Using genome-scale models to predict biological capabilities. *Cell*, 161(5):971–987, 2015.

[14] R. Samaga and S. Klamt. Modeling approaches for qualitative and semi-quantitative analysis of cellular signaling networks. *Cell communication and signaling*, 11(1):43, 2013.

[15] A. Masoudi-Nejad, G. Bidkhori, S. H. Ashtiani, A. Najafi, J. H. Bozorgmehr, and E. Wang. Cancer systems biology and modeling: Microscopic scale and multiscale approaches. In *Seminars in cancer biology*, volume 30, pages 60–69. Elsevier, 2015.

[16] J. L. Reed, T. R. Patel, K. H. Chen, A. R. Joyce, M. K. Applebee, C. D. Herring, O. T. Bui, E. M. Knight, S. S. Fong, and B. O. Palsson. Systems approach to refining genome annotation. *Proceedings of the National Academy of Sciences*, 103(46):17480–17484, 2006.

[17] I. Thiele, N. Swainston, R. M. Fleming, A. Hoppe, S. Sahoo, M. K. Aurich, H. Haraldsdottir, M. L. Mo, O. Rolfsson, M. D. Stobbe, et al. A community-driven global reconstruction of human metabolism. *Nature biotechnology*, 31(5):419–425, 2013.

[18] B. Szappanos, K. Kovács, B. Szamecz, F. Honti, M. Costanzo, A. Baryshnikova, G. Gelius-Dietrich, M. J. Lercher, M. Jelasity, C. L. Myers, et al. An integrated approach to characterize genetic interaction networks in yeast metabolism. *Nature genetics*, 43(7):656–662, 2011.

[19] T. Gaasterland and C. W. Sensen. Magpie: automated genome interpretation. *Trends in Genetics*, 12(2):76–78, 1996.

[20] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70% accuracy. *Journal of molecular biology*, 232(2):584–599, 1993.

[21] B. Rost, C. Sander, R. Casadio, and P. Fariselli. Transmembrane helices predicted at 95% accuracy. *Protein Science*, 4(3):521–533, 1995.

[22] M. Scharf, R. Schneider, G. Casari, P. Bork, A. Valencia, C. Ouzounis, and C. Sander. Genequiz: a workbench for sequence analysis. In *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology, ISMB-94*, pages 348–353. AAAI Press, 1994.

[23] R. D. Fleischmann, M. D. Adams, O. White, R. A. Clayton, et al. Whole-genome random sequencing and assembly of haemophilus influenzae rd. *Science*, 269(5223):496, 1995.

[24] A. Lupas. Predicting coiled-coil regions in proteins. *Current opinion in structural biology*, 7(3):388–393, 1997.

[25] J. Tamames, G. Casari, C. Ouzounis, and A. Valencia. Conserved clusters of functionally related genes in two bacterial genomes. *Journal of molecular evolution*, 44(1):66–73, 1997.

[26] Č. Venclovas, A. Zemla, K. Fidelis, and J. Moult. Comparison of performance in successive casp experiments. *Proteins: Structure, Function, and Bioinformatics*, 45(S5):163–170, 2001.

[27] M. P. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proceedings of the National Academy of Sciences*, 97(1):262–267, 2000.

[28] T. Van Opijnen, K. L. Bodi, and A. Camilli. Tn-seq: high-throughput parallel sequencing for fitness and genetic interaction studies in microorganisms. *Nature methods*, 6(10):767–772, 2009.

[29] M. K. Kerr, M. Martin, and G. A. Churchill. Analysis of variance for gene expression microarray data. *Journal of computational biology*, 7(6):819–837, 2000.

[30] D. A. Lashkari, J. L. DeRisi, J. H. McCusker, A. F. Namath, C. Gentile, S. Y. Hwang, P. O. Brown, and R. W. Davis. Yeast microarrays for genome wide parallel genetic and gene expression analysis. *Proceedings of the National Academy of Sciences*, 94(24):13057–13062, 1997.

[31] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, et al. The sequence of the human genome. *science*, 291(5507):1304–1351, 2001.

[32] G. Chen, C. Wang, and T. Shi. Overview of available methods for diverse rna-seq data analyses. *Science China Life Sciences*, 54(12):1121–1128, 2011.

[33] L. Kearney and S. W. Horsley. Molecular cytogenetics in haematological malignancy: current technology and future prospects. *Chromosoma*, 114(4):286–294, 2005.

[34] Z. Wang, M. Gerstein, and M. Snyder. Rna-seq: a revolutionary tool for transcriptomics. *Nature reviews genetics*, 10(1):57–63, 2009.

[35] J. Clarke, H.-C. Wu, L. Jayasinghe, A. Patel, S. Reid, and H. Bayley. Continuous base identification for single-molecule nanopore dna sequencing. *Nature nanotechnology*, 4(4):265–270, 2009.

[36] E. S. Lander. The heroes of crispr. *Cell*, 164(1):18–28, 2016.

[37] H. Ledford et al. The unsung heroes of crispr. *Nature*, 535(7612):342–344, 2016.

[38] F. Pinzari, A. Ceci, N. Abu-Samra, L. Canfora, O. Maggi, and A. Persiani. Phenotype microarrayâĎć system in the study of fungal functional diversity and catabolic versatility. *Research in microbiology*, 167(9):710–722, 2016.

[39] P. E. Bourne, V. Bonazzi, M. Dunn, E. D. Green, M. Guyer, G. Komatsoulis, J. Larkin, and B. Russell. The nih big data to knowledge (bd2k) initiative. *Journal of the American Medical Informatics Association*, 22(6):1114–1114, 2015.

[40] R. Kamps, R. D. Brandão, B. J. Bosch, A. D. Paulussen, S. Xanthoulea, M. J. Blok, and A. Romano. Next-generation sequencing in oncology: Genetic diagnosis, risk prediction and cancer classification. *International Journal of Molecular Sciences*, 18(2):308, 2017.

[41] P. Groth, G. Reuter, and S. Thieme. Analysis of genomic data in a cloud computing environment. *Big Data Analytics in Bioinformatics and Healthcare*, page 186, 2014.

[42] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer. CopasiâĂŤa complex pathway simulator. *Bioinformatics*, 22(24):3067–3074, 2006.

[43] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504, 2003.

[44] Y. Matsuoka, A. Funahashi, S. Ghosh, and H. Kitano. Modeling and simulation using celldesigner. *Transcription Factor Regulatory Networks: Methods and Protocols*, pages 121–145, 2014.

[45] T. Spiesser, C. Kühn, M. Krantz, and E. Klipp. The mypop toolbox: Putting yeast stress responses in cellular context on single cell and population scales. *Biotechnology journal*, 11(9):1158–1168, 2016.

[46] E. Petelenz-Kurdziel, C. Kuehn, B. Nordlander, D. Klein, K.-K. Hong, T. Jacobson, P. Dahl, J. Schaber, J. Nielsen, S. Hohmann, et al. Quantitative analysis of glycerol accumulation, glycolysis and growth under hyper osmotic stress. *PLoS Comput Biol*, 9(6):e1003084, 2013.

[47] R. L. Chang, K. Andrews, D. Kim, Z. Li, A. Godzik, and B. O. Palsson. Structural systems biology evaluation of metabolic thermotolerance in escherichia coli. *Science*, 340(6137):1220–1223, 2013.

[48] A. Bordbar, J. M. Monk, Z. A. King, and B. O. Palsson. Constraint-based models predict metabolic and associated cellular functions. *Nature Reviews Genetics*, 15(2):107–120, 2014.

[49] E. J. O'brien, J. A. Lerman, R. L. Chang, D. R. Hyduke, and B. Ø. Palsson. Genome-scale models of metabolism and gene expression extend and refine growth phenotype prediction. *Molecular systems biology*, 9(1):693, 2013.

[50] R. Sapranauskas, G. Gasiunas, C. Fremaux, R. Barrangou, P. Horvath, and V. Siksnys. The streptococcus thermophilus crispr/cas system provides immunity in escherichia coli. *Nucleic acids research*, page gkr606, 2011.

[51] R. Barrangou and L. A. Marraffini. Crispr-cas systems: prokaryotes upgrade to adaptive immunity. *Molecular cell*, 54(2):234–244, 2014.

[52] S. Thieme and P. Groth. Genome fusion detection: a novel method to detect fusion genes from snp-array data. *Bioinformatics*, 29(6):671–677, 2013.

[53] J. Wiedenhoeft, E. Brugel, and A. Schliep. Fast bayesian inference of copy number variants using hidden markov models with wavelet compression. *PLoS Comput Biol*, 12(5):e1004871, 2016.

[54] J. E. Haber. Mating-type genes and mat switching in saccharomyces cerevisiae. *Genetics*, 191(1):33–64, 2012.

[55] E. Blackburn and J. Szostak. The molecular structure of centromeres and telomeres. *Annual review of biochemistry*, 53(1):163–194, 1984.

[56] A. M. Neiman. Sporulation in the budding yeast saccharomyces cerevisiae. *Genetics*, 189(3):737–765, 2011.

[57] D. L. Lindstrom and D. E. Gottschling. The mother enrichment program: a genetic system for facile replicative life span analysis in saccharomyces cerevisiae. *Genetics*, 183(2):413–422, 2009.

[58] R. Kostriken, J. N. Strathern, A. J. Klar, J. B. Hicks, and F. Heffron. A site-specific endonuclease essential for mating-type switching in saccharomyces cerevisiae. *Cell*, 35(1):167–174, 1983.

[59] C. M Winters and H.-L. Chiang. Yeast as a model system to study trafficking of small vesicles carrying signal-less proteins in and out of the cell. *Current Protein and Peptide Science*, 17(8):808–820, 2016.

[60] L. H. Hartwell, J. Culotti, J. R. Pringle, and B. J. Reid. Genetic controlof the cell division cycle in yeast. *Science (Wash. DC)*, 183:46–51, 1974.

[61] A. Taddei and S. M. Gasser. Structure and function in the budding yeast nucleus. *Genetics*, 192(1):107–129, 2012.

[62] S. B. Haase and C. Wittenberg. Topology and control of the cell-cycle-regulated transcriptional circuitry. *Genetics*, 196(1):65–90, 2014.

[63] D. Botstein and G. R. Fink. Yeast: an experimental organism for 21st century biology. *Genetics*, 189(3):695–704, 2011.

[64] R. D. Kornberg. The molecular basis of eukaryotic transcription. *Proceedings of the National Academy of Sciences*, 104(32):12955–12961, 2007.

[65] J. E. Rothman. The machinery and principles of vesicle transport in the cell. *Nature medicine*, 8(10):1059–1063, 2002.

[66] K. Siddle. Molecular basis of signaling specificity of insulin and igf receptors: neglected corners and recent advances. *Frontiers in endocrinology*, 3:34, 2012.

[67] M. Barbieri, M. Bonafè, C. Franceschi, and G. Paolisso. Insulin/igf-i-signaling pathway: an evolutionarily conserved mechanism of longevity from yeast to humans. *American Journal of Physiology-Endocrinology And Metabolism*, 285(5):E1064–E1071, 2003.

[68] P. van der Geer, T. Hunter, and R. A. Lindberg. Receptor protein-tyrosine kinases and their signal transduction pathways. *Annual review of cell biology*, 10(1):251–337, 1994.

[69] B. P. Ceresa, A. W. Kao, S. R. Santeler, and J. E. Pessin. Inhibition of clathrin-mediated endocytosis selectively attenuates specific insulin receptor signal transduction pathways. *Molecular and Cellular Biology*, 18(7):3862–3870, 1998.

[70] G. Wolf, T. Trüb, E. Ottinger, L. Groninga, A. Lynch, M. F. White, M. Miyazaki, J. Lee, and S. E. Shoelson. Ptb domains of irs-1 and shc have distinct but overlapping binding specificities. *Journal of Biological Chemistry*, 270(46):27407–27410, 1995.

[71] K. Siddle. Signalling by insulin and igf receptors: supporting acts and new players. *Journal of molecular endocrinology*, 47(1):R1–R10, 2011.

[72] K. Furukawa and S. Hohmann. Synthetic biology: lessons from engineering yeast mapk signalling pathways. *Molecular microbiology*, 88(1):5–19, 2013.

[73] I. Herskowitz. Map kinase pathways in yeast: for mating and more. *Cell*, 80(2):187–197, 1995.

[74] M. C. Gustin, J. Albertyn, M. Alexander, and K. Davenport. Map kinase pathways in the yeastsaccharomyces cerevisiae. *Microbiology and Molecular biology reviews*, 62(4):1264–1300, 1998.

[75] K. B. Lengeler, R. C. Davidson, C. D'souza, T. Harashima, W.-C. Shen, P. Wang, X. Pan, M. Waugh, and J. Heitman. Signal transduction cascades regulating fungal development and virulence. *Microbiology and Molecular Biology Reviews*, 64(4):746–785, 2000.

[76] S. Hohmann. Osmotic stress signaling and osmoadaptation in yeasts. *Microbiology and molecular biology reviews*, 66(2):300–372, 2002.

[77] L. Bardwell. A walk-through of the yeast mating pheromone response pathway. *Peptides*, 25(9):1465–1476, 2004.

[78] D. E. Levin. Regulation of cell wall biogenesis in saccharomyces cerevisiae: the cell wall integrity signaling pathway. *Genetics*, 189(4):1145–1175, 2011.

[79] P. J. Cullen and G. F. Sprague. The regulation of filamentous growth in yeast. *Genetics*, 190(1):23–49, 2012.

[80] P. J. Cullen. Evaluating the activity of the filamentous growth mitogen-activated protein kinase pathway in yeast. *Cold Spring Harbor Protocols*, 2015(3):pdb–prot085092, 2015.

[81] E. Leberer, D. Y. Thomas, and M. Whiteway. Pheromone signalling and polarized morphogenesis in yeast. *Current opinion in genetics & development*, 7(1):59–66, 1997.

[82] S. Nomoto, N. Nakayama, K.-i. Arai, and K. Matsumoto. Regulation of the yeast pheromone response pathway by g protein subunits. *The EMBO journal*, 9(3):691, 1990.

[83] T. Leeuw, C. Wu, J. D. Schrag, M. Whiteway, D. Y. Thomas, and E. Leberer. Interaction of a g-protein β-subunit with a conserved sequence in ste20/pak family protein kinases. *Nature*, 391(6663):191–195, 1998.

[84] S. Klein, H. Reuveni, and A. Levitzki. Signal transduction by a nondissociable heterotrimeric yeast g protein. *Proceedings of the National Academy of Sciences*, 97(7):3219–3223, 2000.

[85] M. Whiteway. Association of the yeast pheromone response g protein $\beta\gamma$ subunits with the map kinase scaffold ste5p. *Trends in Genetics*, 2(12):47, 1996.

[86] K.-Y. Choi, B. Satterberg, D. M. Lyons, and E. A. Elion. Ste5 tethers multiple protein kinases in the map kinase cascade required for mating in s. cerevisiae. *Cell*, 78(3):499–512, 1994.

[87] F. van Drogen, S. M. OâĂŹRourke, V. M. Stucke, M. Jaquenoud, A. M. Neiman, and M. Peter. Phosphorylation of the mekk ste11p by the pak-like kinase ste20p is required for map kinase signaling in vivo. *Current Biology*, 10(11):630–639, 2000.

[88] R. E. Lamson, S. Takahashi, M. J. Winters, and P. M. Pryciak. Dual role for membrane localization in yeast map kinase cascade activation and its contribution to signaling fidelity. *Current biology*, 16(6):618–623, 2006.

[89] W. R. Burack and A. S. Shaw. Signal transduction: hanging on a scaffold. *Current opinion in cell biology*, 12(2):211–216, 2000.

[90] M. Ptashne and A. Gann. Imposing specificity on kinases. *Science*, 299(5609):1025–1027, 2003.

[91] F. Chang and I. Herskowitz. Identification of a gene necessary for cell cycle arrest by a negative growth factor of yeast: Far1 is an inhibitor of a g1 cyclin, cln2. *Cell*, 63(5):999–1011, 1990.

[92] Y. Shimada, M.-P. Gulli, and M. Peter. Nuclear sequestration of the exchange factor cdc24 by far1 regulates cell polarity during yeast mating. *Nature cell biology*, 2(2):117–124, 2000.

[93] A. Nern and R. A. Arkowitz. A cdc24p-far1p-g$\beta\gamma$ protein complex required for yeast orientation during mating. *The Journal of cell biology*, 144(6):1187–1202, 1999.

[94] S. Henchoz, Y. Chi, B. Catarin, I. Herskowitz, R. J. Deshaies, and M. Peter. Phosphorylation-and ubiquitin-dependent degradation of the cyclin-dependent kinase inhibitor far1p in budding yeast. *Genes & Development*, 11(22):3046–3060, 1997.

[95] M. Peter and I. Herskowitz. Direct inhibition of the yeast cyclin-dependent kinase cdc28-cln by far1. *Science*, 265(5176):1228–1232, 1994.

[96] M. Peter, A. Gartner, J. Horecka, G. Ammerer, and I. Herskowitz. Far1 links the signal transduction pathway to the cell cycle machinery in yeast. *Cell*, 73(4):747–760, 1993.

[97] E. Elion, B. Satterberg, and J. Kranz. Fus3 phosphorylates multiple components of the mating signal transduction cascade: evidence for ste12 and far1. *Molecular biology of the cell*, 4(5):495–510, 1993.

[98] B. Errede and G. Ammerer. Ste12, a protein involved in cell-type-specific transcription and signal transduction in yeast, is part of protein-dna complexes. *Genes & development*, 3(9):1349–1361, 1989.

[99] C. J. Roberts, B. Nelson, M. J. Marton, R. Stoughton, M. R. Meyer, H. A. Bennett, Y. D. He, H. Dai, W. L. Walker, T. R. Hughes, et al. Signaling and circuitry of multiple mapk pathways revealed by a matrix of global gene expression profiles. *Science*, 287(5454):873–880, 2000.

[100] S. Chou, S. Lane, and H. Liu. Regulation of mating and filamentation genes by two distinct ste12 complexes in saccharomyces cerevisiae. *Molecular and Cellular Biology*, 26(13):4794–4805, 2006.

[101] R. L. Roberts and G. R. Fink. Elements of a single map kinase cascade in saccharomyces cerevisiae mediate two developmental programs in the same cell type: mating and invasive growth. *Genes & development*, 8(24):2974–2985, 1994.

[102] J. M. Johnson, M. Jin, and D. J. Lew. Symmetry breaking and the establishment of cell polarity in budding yeast. *Current opinion in genetics & development*, 21(6):740–746, 2011.

[103] E. de Nadal, P. M. Alepuz, and F. Posas. Dealing with osmostress through map kinase activation. *EMBO reports*, 3(8):735–740, 2002.

[104] E. Winter and M. C. Gustin. An osmosensing signal transduction pathway in yeast. *Cell*, 50:495, 1987.

[105] T. Maeda, S. M. Wurgler-Murphy, and H. Saito. A two-component system that regulates an osmosensing map kinase cascade in yeast. *Nature*, 369(6477):242, 1994.

[106] F. Posas, S. M. Wurgler-Murphy, T. Maeda, E. A. Witten, T. C. Thai, and H. Saito. Yeast hog1 map kinase cascade is regulated by a multistep phosphorelay mechanism in the sln1–ypd1–ssk1 âĂIJtwo-componentâĂİ osmosensor. *Cell*, 86(6):865–875, 1996.

[107] F. Posas and H. Saito. Activation of the yeast ssk2 map kinase kinase kinase by the ssk1 two-component response regulator. *The EMBO journal*, 17(5):1385–1394, 1998.

[108] K. Tatebayashi, K. Tanaka, H.-Y. Yang, K. Yamamoto, Y. Matsushita, T. Tomida, M. Imai, and H. Saito. Transmembrane mucins hkr1 and msb2 are putative osmosensors in the sho1 branch of yeast hog pathway. *The EMBO journal*, 26(15):3521–3533, 2007.

[109] V. Reiser, S. M. Salah, and G. Ammerer. Polarized localization of yeast pbs2 depends on osmostress, the membrane protein sho1 and cdc42. *Nature Cell Biology*, 2(9):620–627, 2000.

[110] A. Zarrinpar, R. P. Bhattacharyya, M. P. Nittler, and W. A. Lim. Sho1 and pbs2 act as coscaffolds linking components in the yeast high osmolarity map kinase pathway. *Molecular cell*, 14(6):825–832, 2004.

[111] D. C. Raitt, F. Posas, and H. Saito. Yeast cdc42 gtpase and ste20 pak-like kinase regulate sho1-dependent activation of the hog1 mapk pathway. *The EMBO journal*, 19(17):4623–4631, 2000.

[112] H. Saito and K. Tatebayashi. Regulation of the osmoregulatory hog mapk cascade in yeast. *The Journal of Biochemistry*, 136(3):267–272, 2004.

[113] P. Ferrigno, F. Posas, D. Koepp, H. Saito, and P. A. Silver. Regulated nucleo/cytoplasmic exchange of hog1 mapk requires the importin β homologs nmd5 and xpo1. *The EMBO journal*, 17(19):5606–5614, 1998.

[114] M. Rep, V. Reiser, U. Gartner, J. M. Thevelein, S. Hohmann, G. Ammerer, and H. Ruis. Osmotic stress-induced gene expression in saccharomyces cerevisiae requires msn1p and the novel nuclear factor hot1p. *Molecular and cellular biology*, 19(8):5474–5485, 1999.

[115] M. Rep, M. Krantz, J. M. Thevelein, and S. Hohmann. The transcriptional response of saccharomyces cerevisiae to osmotic shock hot1p and msn2p/msn4p are required for the induction of subsets of high osmolarity glycerol pathway-dependent genes. *Journal of Biological Chemistry*, 275(12):8290–8300, 2000.

[116] C. Ferreira, F. van Voorst, A. Martins, L. Neves, R. Oliveira, M. C. Kielland-Brandt, C. Lucas, and A. Brandt. A member of the sugar transporter family, stl1p is the glycerol/h+ symporter in saccharomyces cerevisiae. *Molecular Biology of the Cell*, 16(4):2068–2076, 2005.

[117] H. Dihazi, R. Kessler, and K. Eschrich. High osmolarity glycerol (hog) pathway-induced phosphorylation and activation of 6-phosphofructo-2-kinase are essential for glycerol accumulation and yeast cell proliferation under hyperosmotic stress. *Journal of Biological Chemistry*, 279(23):23961–23968, 2004.

[118] M. Thorsen, Y. Di, C. Tängemo, M. Morillas, D. Ahmadpour, C. Van der Does, A. Wagner, E. Johansson, J. Boman, F. Posas, et al. The mapk hog1p modulates fps1p-dependent arsenite uptake and tolerance in yeast. *Molecular biology of the cell*, 17(10):4400–4410, 2006.

[119] H. Saito and F. Posas. Response to hyperosmotic stress. *Genetics*, 192(2):289–318, 2012.

[120] M. L. Blinov, J. R. Faeder, B. Goldstein, and W. S. Hlavacek. Bionetgen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics*, 20(17):3289–3291, 2004.

[121] C.-F. Tiger, F. Krause, G. Cedersund, R. Palmér, E. Klipp, S. Hohmann, H. Kitano, and M. Krantz. A framework for mapping, visualisation and automatic model creation of signal-transduction networks. *Molecular systems biology*, 8(1):578, 2012.

[122] J. C. Romers, S. Thieme, U. Münzner, and M. Krantz. Using rxncon to develop rule based models. *submitted*, 2017.

[123] I. Thiele and B. Ø. Palsson. A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nature protocols*, 5(1):93–121, 2010.

[124] J. C. Romers and M. Krantz. rxncon 2.0: a language for executable molecular systems biology. *bioRxiv*, page 107136, 2017.

[125] N. Juty, N. Le Novere, and C. Laibe. Identifiers. org and miriam registry: community resources to provide persistent identification. *Nucleic acids research*, 40(D1):D580–D586, 2012.

[126] J. M. Cherry, E. L. Hong, C. Amundsen, R. Balakrishnan, G. Binkley, E. T. Chan, K. R. Christie, M. C. Costanzo, S. S. Dwight, S. R. Engel, et al. Saccharomyces genome database: the genomics resource of budding yeast. *Nucleic acids research*, page gkr1029, 2011.

[127] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, et al. The systems biology markup language (sbml): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[128] E. Demir, M. P. Cary, S. Paley, K. Fukuda, C. Lemer, I. Vastrik, G. Wu, P. D'eustachio, C. Schaefer, J. Luciano, et al. The biopax community standard for pathway data sharing. *Nature biotechnology*, 28(9):935–942, 2010.

[129] N. Le Novere, M. Hucka, H. Mi, S. Moodie, F. Schreiber, A. Sorokin, E. Demir, K. Wegner, M. I. Aladjem, S. M. Wimalaratne, et al. The systems biology graphical notation. *Nature biotechnology*, 27(8):735–741, 2009.

[130] N. Le Novère, A. Finney, M. Hucka, U. S. Bhalla, F. Campagne, J. Collado-Vides, E. J. Crampin, M. Halstead, E. Klipp, P. Mendes, et al. Minimum information requested in the annotation of biochemical models (miriam). *Nature biotechnology*, 23(12):1509–1515, 2005.

[131] D. Waltemath, R. Adams, D. A. Beard, F. T. Bergmann, U. S. Bhalla, R. Britten, V. Chelliah, M. T. Cooling, J. Cooper, E. J. Crampin, et al. Minimum information about a simulation experiment (miase). *PLoS Comput Biol*, 7(4):e1001122, 2011.

[132] M. Rother, U. Münzner, S. Thieme, and M. Krantz. Information content and scalability in signal transduction network reconstruction formats. *Molecular BioSystems*, 9(8):1993–2004, 2013.

[133] M. J. Herrgård, N. Swainston, P. Dobson, W. B. Dunn, K. Y. Arga, M. Arvas, N. Blüthgen, S. Borger, R. Costenoble, M. Heinemann, et al. A consensus yeast metabolic network reconstruction obtained from a community approach to systems biology. *Nature biotechnology*, 26(10):1155–1160, 2008.

[134] J. R. Faeder, M. L. Blinov, B. Goldstein, and W. S. Hlavacek. Rule-based modeling of biochemical networks. *Complexity*, 10(4):22–41, 2005.

[135] V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine. Rule-based modelling of cellular signalling. In *International Conference on Concurrency Theory*, pages 17–41. Springer, 2007.

[136] T. Lubitz, N. Welkenhuysen, S. Shashkova, L. Bendrioua, S. Hohmann, E. Klipp, and M. Krantz. Network reconstruction and validation of the snf1/ampk pathway in bakerâĂŹs yeast based on a comprehensive literature review. *npj Systems Biology and Applications*, 1:15007, 2015.

[137] S. Hohmann. Control of high osmolarity signalling in the yeast saccharomyces cerevisiae. *FEBS letters*, 583(24):4025–4029, 2009.

[138] Y. Murakami, K. Tatebayashi, and H. Saito. Two adjacent docking sites in the yeast hog1 mitogen-activated protein (map) kinase differentially interact with the pbs2 map kinase kinase and the ptp2 protein tyrosine phosphatase. *Molecular and cellular biology*, 28(7):2481–2494, 2008.

[139] P. M. Alepuz, E. de Nadal, M. Zapater, G. Ammerer, and F. Posas. Osmostress-induced transcription by hot1 depends on a hog1-mediated recruitment of the rna pol ii. *The EMBO journal*, 22(10):2433–2442, 2003.

[140] S. M. Wurgler-Murphy, T. Maeda, E. A. Witten, and H. Saito. Regulation of the saccharomyces cerevisiae hog1 mitogen-activated protein kinase by the ptp2 and ptp3 protein tyrosine phosphatases. *Molecular and Cellular Biology*, 17(3):1289–1297, 1997.

[141] R. G. Jeroslow and J. Wang. Solving propositional satisfiability problems. *Annals of mathematics and Artificial Intelligence*, 1(1):167–187, 1990.

[142] A. Biere. Picosat essentials. *Journal on Satisfiability, Boolean Modeling and Computation*, 4:75–97, 2008.

[143] C. Drake. Python library for electronic design automation (pyeda). Accessed: 2017-04-27 (https://pypi.python.org/pypi/pyeda).

[144] M. Flöttmann, F. Krause, E. Klipp, and M. Krantz. Reaction-contingency based bipartite boolean modelling. *BMC systems biology*, 7(1):58, 2013.

[145] L. A. Chylek, B. Hu, M. L. Blinov, T. Emonet, J. R. Faeder, B. Goldstein, R. N. Gutenkunst, J. M. Haugh, T. Lipniacki, R. G. Posner, et al. Guidelines for visualizing and annotating rule-based models. *Molecular BioSystems*, 7(10):2779–2795, 2011.

[146] H.-M. Kaltenbach, S. Constantinescu, J. Feigelman, and J. Stelling. Graph-based decomposition of biochemical reaction networks into monotone subsystems. In *International Workshop on Algorithms in Bioinformatics*, pages 139–150. Springer, 2011.

[147] T. M. Thomson. Yeast pheromone model. Accessed: 2017-04-20 (http://yeastpheromonemodel.org).

[148] T. M. Thomson, K. R. Benjamin, A. Bush, T. Love, D. Pincus, O. Resnekov, C. Y. Richard, A. Gordon, A. Colman-Lerner, D. Endy, et al. Scaffold number in yeast signaling system sets tradeoff between system output and dynamic range. *Proceedings of the National Academy of Sciences*, 108(50):20265–20270, 2011.

[149] A. P. Oliveira, C. Ludwig, P. Picotti, M. Kogadeeva, R. Aebersold, and U. Sauer. Regulation of yeast central metabolism by enzyme phosphorylation. *Molecular Systems Biology*, 8(1):623, 2012.

[150] S. A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467, 1969.

[151] R. Thomas. Boolean formalization of genetic control circuits. *Journal of theoretical biology*, 42(3):563–585, 1973.

[152] Y. Xiao. A tutorial on analysis and simulation of boolean gene regulatory network models. *Current genomics*, 10(7):511–525, 2009.

[153] D. Cheng and H. Qi. State–space analysis of boolean networks. *IEEE Transactions on Neural Networks*, 21(4):584–594, 2010.

[154] M. I. Davidich and S. Bornholdt. Boolean network model predicts cell cycle sequence of fission yeast. *PloS one*, 3(2):e1672, 2008.

[155] M. I. Davidich and S. Bornholdt. Boolean network model predicts knockout mutant phenotypes of fission yeast. *PLoS One*, 8(9):e71786, 2013.

[156] G. A. Ruz, T. Timmermann, J. Barrera, and E. Goles. Neutral space analysis for a boolean network model of the fission yeast cell cycle network. *Biological research*, 47(1):64, 2014.

[157] T. Handorf and E. Klipp. Modeling mechanistic biological networks: an advanced boolean approach. *Bioinformatics*, 28(4):557–563, 2012.

[158] K. Kolczyk, R. Samaga, H. Conzelmann, S. Mirschel, and C. Conradi. The process-interaction-model: a common representation of rule-based and logical models allows studying signal transduction on different levels of detail. *BMC bioinformatics*, 13(1):251, 2012.

[159] R.-S. Wang, A. Saadatpour, and R. Albert. Boolean modeling in systems biology: an overview of methodology and applications. *Physical biology*, 9(5):055001, 2012.

[160] C. Müssel, M. Hopfensitz, and H. A. Kestler. BoolnetâĂŤan r package for generation, reconstruction and analysis of boolean networks. *Bioinformatics*, 26(10):1378–1380, 2010.

[161] E. Klipp, B. Nordlander, R. Krüger, P. Gennemark, and S. Hohmann. Integrative model of the response of yeast to osmotic shock. *Nature biotechnology*, 23(8):975–982, 2005.

[162] N. Le Novere and L. Endler. Using chemical kinetics to model biochemical pathways. *In Silico Systems Biology*, pages 147–167, 2013.

[163] D. T. Gillespie, A. Hellander, and L. R. Petzold. Perspective: Stochastic algorithms for chemical kinetics. *The Journal of chemical physics*, 138(17):05B201_1, 2013.

[164] N. M. Borisov, A. S. Chistopolsky, J. R. Faeder, and B. N. Kholodenko. Domain-oriented reduction of rule-based network models. *IET systems biology*, 2(5):342–351, 2008.

[165] H. Conzelmann, D. Fey, and E. D. Gilles. Exact model reduction of combinatorial reaction networks. *BMC Systems Biology*, 2(1):78, 2008.

[166] L. A. Harris, J. S. Hogg, J.-J. Tapia, J. A. Sekar, S. Gupta, I. Korsunsky, A. Arora, D. Barua, R. P. Sheehan, and J. R. Faeder. Bionetgen 2.2: advances in rule-based modeling. *Bioinformatics*, page btw469, 2016.

[167] L. A. Chylek, L. A. Harris, J. R. Faeder, and W. S. Hlavacek. Modeling for (physical) biologists: an introduction to the rule-based approach. *Physical biology*, 12(4):045007, 2015.

[168] W. S. Hlavacek, J. R. Faeder, M. L. Blinov, R. G. Posner, M. Hucka, and W. Fontana. Rules for modeling signal-transduction systems. *Sci STKE*, 2006(344):re6, 2006.

[169] S. Eker, M. Knapp, K. Laderoute, P. Lincoln, J. Meseguer, and K. Sonmez. Pathway logic: Symbolic analysis of biological signaling. In *Pacific Symposium on Biocomputing*, volume 7, pages 400–412, 2002.

[170] W. S. Hlavacek, J. R. Faeder, M. L. Blinov, A. S. Perelson, and B. Goldstein. The complexity of complexes in signal transduction. *Biotechnology and bioengineering*, 84(7):783–794, 2003.

[171] M. Koschorreck, H. Conzelmann, S. Ebert, M. Ederer, and E. D. Gilles. Reduced modeling of signal transduction–a modular approach. *BMC bioinformatics*, 8(1):336, 2007.

[172] N. W. Lemons, B. Hu, and W. S. Hlavacek. Hierarchical graphs for rule-based modeling of biochemical systems. *BMC bioinformatics*, 12(1):45, 2011.

[173] M. L. Blinov, J. Yang, J. R. Faeder, and W. S. Hlavacek. Graph theory for rule-based modeling of biochemical networks. In *Transactions on Computational Systems Biology VII*, pages 89–106. Springer, 2006.

[174] M. Löwe. Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109(1):181–224, 1993.

[175] J. R. Faeder, M. L. Blinov, and W. S. Hlavacek. Rule-based modeling of biochemical systems with bionetgen. *Systems biology*, pages 113–167, 2009.

[176] B. Goldstein, J. R. Faeder, W. S. Hlavacek, M. L. Blinov, A. Redondo, and C. Wofsy. Modeling the early signaling events mediated by fcεri. *Molecular Immunology*, 38(16):1213–1219, 2002.

[177] J. R. Faeder, W. S. Hlavacek, I. Reischl, M. L. Blinov, H. Metzger, A. Redondo, C. Wofsy, and B. Goldstein. Investigation of early events in fcεri-mediated signaling using a detailed mathematical model. *The Journal of Immunology*, 170(7):3769–3781, 2003.

[178] M. L. Blinov and I. I. Moraru. Leveraging modeling approaches: reaction networks and rules. In *Advances in Systems Biology*, pages 517–530. Springer, 2012.

[179] M. Ruth and B. Hannon. Modeling dynamic biological systems. In *Modeling Dynamic Biological Systems*, pages 3–27. Springer, 1997.

[180] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of computational physics*, 22(4):403–434, 1976.

[181] M. W. Sneddon, J. R. Faeder, and T. Emonet. Efficient modeling, simulation and coarse-graining of biological complexity with nfsim. *Nature methods*, 8(2):177–183, 2011.

[182] J. A. Sekar and J. R. Faeder. Rule-based modeling of signal transduction: a primer. *Computational Modeling of Signaling Networks*, pages 139–218, 2012.

[183] L. A. Chylek, L. A. Harris, C.-S. Tung, J. R. Faeder, C. F. Lopez, and W. S. Hlavacek. Rule-based modeling: a computational approach for studying biomolecular site dynamics in cell signaling systems. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 6(1):13–36, 2014.

[184] J. Yang, M. I. Monine, J. R. Faeder, and W. S. Hlavacek. Kinetic monte carlo method for rule-based modeling of biochemical networks. *Physical Review E*, 78(3):031910, 2008.

[185] L. A. Chylek, E. C. Stites, R. G. Posner, and W. S. Hlavacek. Innovations of the rule-based modeling approach. In *Systems Biology*, pages 273–300. Springer, 2013.

[186] T. Lubitz, J. Hahn, F. T. Bergmann, E. Noor, E. Klipp, and W. Liebermeister. Sbtab: a flexible table format for data exchange in systems biology. *Bioinformatics*, page btw179, 2016.

[187] T. Mori, M. Flöttmann, M. Krantz, T. Akutsu, and E. Klipp. Stochastic simulation of boolean rxncon models: towards quantitative analysis of large signaling networks. *BMC systems biology*, 9(1):45, 2015.

[188] U. Münzner, T. Lubitz, E. Klipp, and M. Krantz. Toward genome-scale models of signal transduction networks. *Systems Biology*, 6, 2017.

[189] S. Thieme, J. C. Romers, U. Muenzner, and M. Krantz. Bipartite boolean modelling-a method for mechanistic simulation and validation of large-scale signal transduction networks. *bioRxiv*, page 107235, 2017.

List of Figures

List of Tables

List of Abbreviations

Abbreviation Full	name
bBM	bipartite Boolean Model
CNF	conjunctive normal form
DNA	Deoxyribonucleic acid
DNF	disjunctive normal form
Dig	Down-regulator of invasive growth
xgmml	eXtensible Graph Markup and Modeling Language
Far	Factor arrest
Fus	Fusion
Gpa1	G-protein alpha subunit
Hog	High Osmolarity Glycerol response
Hot	High-Osmolarity-induced Transcription
HO	HOmothallic switching endonuclease
Kss1	Kinase-suppressor of Sst2
MEK	MAPK/ERK kinase
MAT	Mating type locus
MAPK	mitogen-activated protein kinase
Msb2	Multicopy Suppression of a Budding defect
Msg5	Multicopy suppressor of GPA1 deletion
Pbs	Polymyxin B Sensitivity

Ptp	Protein tyrosine phosphatase
rxncon	reaction-contingency
Ste	Sterile
Sst	Supersensitive
Ssk	Suppressor of Sensor Kinase
sln1	Synthetic Lethal of N-end rule
Sho1	Synthetic, High Osmolarity-sensitive
Thr	Threonin
Thy	Tyrosin
Ypd	tYrosine Phosphatase Dependent
UC	undifined catalyst

Glossary

Saccharomyces cerevisiae

Alias: yeast, budding yeast
Description: Simple eukaryotic model organism.

Bond name

Description: Defined by a "!" followed by a number.

Boolean component expression

Alias: component expression
Description: Boolean expression of all Boolean state nodes representing rxncon states sharing the same rxncon component.

Boolean system

Alias: Boolean network
Description: Collection of Boolean nodes and Boolean update functions.

Boolean state nodes

Alias: state node
Description: Boolean node representing a rxncon state.

Boolean state vector

Description: A vector, describing all Boolean states at time t.

Boolean state

Description: Activity status of a Boolean node defined by a logical value.

Boolean update function

Alias: Boolean function, update function
Description: Function, describing the transition of one Boolean state into another.

Boolean component

Description: Represented by Boolean component expression.

Boolean-contingency term

Alias: Boolean statement, rxncon complex
Description: Described by rxncon states and/or nested Boolean-contingency terms concatenated by Boolean operators; representing biological complexes.
For instance:
`Pbs2_p+_Hog1_[(Thr174)]; ! <pbs2-active>`
`<pbs2-active>; AND Pbs2_[(Thr518)]-{p}`
`<pbs2-active>; AND Pbs2_[(Ser514)]-{p}`.

Component

Description: Defines the Name of the molecule and describes molecule type.
For instance: `Hog1`, `Hog1mRNA`, `Hog1Gene`.

Component level

Description: Specification without Locus information; the resolution of a molecule (resolution of a molecule is discriminated between component, domain and residue level).
For instance: `Pbs2`.

Contingency type

Description: Relation between Target and Effector.
For instance: `!`, `x`, `?`,`0`, `k+`, `k-`, `AND`, `OR`, `NOT`.

Contingency list

Description: List containing contextual constraints..

Degradation reaction

Description: rxncon reaction describing the degradation of a molecule.
For instance: `A_deg_B`.

Domain level

Description: Specification with domain information on the Locus information; the resolution of a molecule (resolution of a molecule is discriminated between component, domain and residue level).
For instance: `Pbs2`.

Effector

Description: A Boolean statement, elemental state or input state influencing a rxncon reaction.
For instance: `Pbs2_p+_Hog1_[(Thr174)]; ! Pbs2_[(Ser514)]-{p}`, where Pbs2_[(Ser514)]-{p} is the Effector.

Elemental resolution

Description: rxncon reaction describing the degradation of a molecule..

Elemental state

Description: rxncon state defined on an elemental resolution.
For instance: `Hog1_[(Thr174)]-{p}`, Pbs2_[HBD1]–Hog1_[CD].

Elemental reaction

Description: rxncon reaction defined on an elemental resolution.
For instance: `Pbs2_p+_Hog1_[(Thr174)]`.

Empirical data

Alias: experimental data
Description: Data obtained by experiments..

Empty-binding state

Description: rxncon state, representing a molecule with an empty binding domain.
For instance: `Pbs2_[HBD1]-0`.

Fully-neutral state

Description: all neutral states of a certain component, which describes a microstate in its fully neutral configuration (not modified, not bound).

Generic component state node

Description: Boolean state node that do not carry any further structure, e.g. modification or binding.

Generic update function

Description: A generalised Boolean update function.

Global state node

Alias: input state node
Description: represents global input state.

Global reaction node

Alias: output reaction node
Description: represents global output reaction.

Graphical-quantity node

Description: Represents a global quantity, e.g. input state and output reaction within a visualisation..

Graphical-Boolean node

Description: Represents a Boolean operators within a visualisation..

Graphical-state node

Description: Represents an elemental states within a visualisation..

Graphical-reaction node

Description: Represents an elemental reaction within a visualisation..

Graphical-residue node

Description: Represents the residue locus information of a specification within a visualisation..

Graphical-domain node

Description: Represents the domain locus information of a specification within a visualisation..

Graphical-component node

Description: Represents a component within a visualisation; Within a species-reaction or regulatory graph this node only appears, if the component do not contain any rxncon states but get regulated by the rxncon system..

Handshake molecule

Description: Molecules defined within a rxncon complex that enables a connection to the Target rxncon reaction. Each Boolean contingency term has its own requirements on the molecules..

Homo-dimer

Description: A rxncon interaction state describing the interaction between two molecules of the same name.
For instance: `A_[x]-A_[y]`.

Input state

Alias: global state
Description: rxncon state with no specification, representing a global quantity.
For instance: `Sln1_ap+_Sln1; x [Turgor]`.

Interaction state

Description: rxncon state with a pair of specifications bound to each other.
For instance: `Pbs2_[HBD1]-Hog1_[CD]`.

Interaction reaction

Description: rxncon reaction describing the interaction between two molecules.
For instance: `Pbs2_[HBD1]_ppi+_Hog1_[CD]`.

Locus

Description: Defines the location (structural element) on a molecule (e.g. domain, residue) as well as the resolution a molecule is defined on.
For instance: `Pbs2_[HBD1]`, `Hog1_[(Thr174)]`.

Mitogen-activated protein kinase

Description: Sequential activation of three kinases (MEKK, MEK, MAPK).

Modification state

Description: rxncon state, representing a molecule modification.
For instance: `Hog1_[(Thr174)]-{p}`.

Molecular component

Description: Molecule containing all sites and states defined in the rule-based system.

Molecule

Alias: agent, species
Description: Collection of biochemical macromolecules grouped by structural elements (macrostates).

Molecule type

Description: Type of the molecule e.g., Protein.

Motif

Alias: pattern
Description: One or more sites and/or states within a molecule.

Namespace

Description: Concept to organize specifications, allowing the reuse of names in different context..

Neutral state

Description: rxncon state without any modification or binding.
For instance: `Hog1_[(Thr174)]-{0}`, `Pbs2_[HBD1]-0`.

Non-elemental state

Description: rxncon state not on an elemental resolution.
For instance: `Hog1-{p}`, `Pbs2-Hog1`.

Object

Description: Specifications within a rxncon reaction..

Observable

Description: Output function that relates molecules of the model to experimental data as cumulative quantities, e.g. the concentration of a phosphorylated protein.

Predicate

Description: Reaction type within a rxncon reaction..

Product motif

Description: One or more sites and/or states within a molecule on the right-hand side of the rule.

Rate law

Description: Describes the relation between the concentration of the reactants and the rate of the reaction.

Reactant

Description: In context of rule-based modelling: Molecule on the left-hand side of a rule; In context of a rxncon system: A specification of a rxncon reaction..

Reactant motif

Description: One or more sites and/or states within a molecule on the left-hand side of the rule.

Reaction centre

Description: Describing the molecules that change during the reaction, defined by rxncon reaction.

Reaction rule

Alias: reaction, rule
Description: Describing the transition of molecules, representing one or more biomolecular reactions.

Reaction node

Description: Boolean node representing a rxncon reaction.

Reaction type

Description: type of a rxncon reaction, e.g phoshorylation.
For instance: `ppi+`, `p+`, `p-`.

Reaction layer

Description: The collection of all rxncon reactions within the rxncon system defines the reaction layer..

Reaction list

Description: List containing rxncon reactions..

Reactive site

Description: A site defined within a molecule which is functionally important for the rule.

Regulatory layer

Description: Regulatory mechanisms of a reconstructed rxncon network..

Residue level

Description: Specification with residue information on the Locus information; the resolution of a molecule (resolution of a molecule is discriminated between component, domain and residue level).
For instance: `Pbs2`.

Rule-based model

Alias: rule-based system
Description: Collection of rules.

rxncon state property

Description: modification or binding of a rxncon state..

rxncon states

Alias: macroscopic state, elemental state, non-elemental state
Description: Defines a state within the rxncon system; describes only one property of a biological state.
For instance: `Pbs2_[HBD1]-Hog1_[CD]`, `Hog1_[(Thr174)]-{p}`.

rxncon reaction

Alias: reaction term
Description: Denotes which property of a molecule changes, without resorting the microstate description.
For instance: `Pbs2_[HBD1]_ppi+_Hog1_[CD]`.

rxncon system

Alias: rxncon model, rxncon network
Description: Summarises the knowledge about the mechanistic processes of reconstructed pathways described by the rxncon language. Collection of rxncon reactions and contingencies..

rxncon language

Description: Formal language for reconstructing biological processes; describes a rxncon system..

Self-interaction state

Description: rxncon state, representing a molecule interacting with itself.
For instance: `A_[x]-[y]`.

Site

Alias: component
Description: Structural element describing a molecular domain or molecular residue, represented by the rxncon specification locus.

Site state

Description: Internal state of a site e.g., modification (site P).

Site type

Description: Name of the structural element.

Skeleton term

Description: Defines a component and its rxncon state within the skeleton rule..

Skeleton rule

Description: Defines the semantic of a rxncon reaction..

Solution

Description: In respect to a rxncon system a semantically correct interpretation of a rxncon statement. In respect to a Boolean model a satisfiable Boolean function.

Source state node

Description: representing a rxncon state consumed by a rxncon reaction.

Specification

Description: Building block of rxncon language. Defines the type and the structure of a biological molecule. In context with rxncon reactions and states it corresponds to biological molecule.
For instance: `Hog1_[(Thr174)]`.

Specific state

Alias: microstate
Description: Fully specified biological molecule, e.g. a protein with a specific configuration including all its non-mutually exclusive modifications and bindings.

State node

Description: Boolean node representing a rxncon state.

State space

Description: Describes the space of all possible states of a system.

Structural index

Description: Integer related to a specification.
For instance: `Hog1@1_[(Thr174)]-{p}`.

Structured complex

Description: .
For instance:
`Pbs2_p+_Hog1_[(Thr174)]; ! <pbs2-active>Pbs2@0=Pbs2@1`
`<pbs2-active>; AND Pbs2@1_[(Thr518)]-{p}`
`<pbs2-active>; AND Pbs2@1_[(Ser514)]-{p}`.

Subject

Description: Specifications within a rxncon reaction..

Synthesis reaction

Description: rxncon reaction describing the synthesis of a molecule.
For instance: `A_syn_B`.

Target

Description: Defines the target (rxncon reaction or Boolean statement) which is regulated by a contingency.
For instance: `Pbs2_p+_Hog1_[(Thr174)]; ! Pbs2_[(Ser514)]-{p}`, where `Pbs2_p+_Hog1_[(Thr174)]` is the Target.

Appendix

Overview of proteins of the reconstructed pathways

Table S2: Description of some proteins involved in the human insulin response pathway

Proteins	Description
IR/IGFR	Insulin binding receptors, recruiting and phosphorylation of IRS and Shc
IRS	insulin receptor substrates, triggering the activation of PI3K
Shc	SHC-transforming protein 2, phosphorylated by IR, binding to Grb2/SOS complex
PI3K	phosphoinositide 3-kinase, activating Akt/PKB cascade, activated by IRS
Ras/MAPK	Activating MAPK cascade involved in proliferation and differentiation of the cell
Akt/PKB	Akt/PKB cascade, activated by PI3K, involved in cell survival, cell growth and proliferation
Grb2	growth factor receptor-binding protein 2, binding of SOS, IRS and Shc
SOS	son of sevenless (SOS Ras/Rac guanine nucleotide exchange factor 1), binding to Grb2

Table S3: Description of some proteins involved in the pheromone response pathway in yeast

Proteins	Description
Ste2/3	7-transmembrane-segment, G-protein coupled pheromone receptors; Binds pheromone
Gpa1	G-protein alpha subunit; triggering an adaptive response by releasing from the beta (Ste4) and gamma (Ste18) subunits
Ste4, Ste18	G-protein beta-gamma subunits; activating the signalling branch
Cdc42	Small rho-like G protein; binds to Ste20; through binding activating of the kinase activity of Ste20
Ste5	Pheromone responsive MAPK scaffold protein; binds G-beta subunit as well as MAPK cascade kinases and other
Ste11	MEKK (MEK kinase); activated by Ste20
Ste20	PAK (p21-activated protein kinase); activated by Cdc42; autophosphorylation
Ste7	MEK (MAPK/ERK kinase); activated by Ste11
Kss1, Fus3	MAP kinases; activated by Ste7
Far1	MAPK substrate; inhibits cell-cycle progression, scaffold that binds G-beta subunit, Cdc24 and other
Cdc24	Guanine nucleotide exchange factor (GEF) for Cdc42; Involved in polarity establishment
Dig1, Dig2	MAPK substrates, repressors of Ste12 transcriptional activity
Ste12	MAPK substrate, Transcription factor

rxncon input formats

The rxncon tool accepts two different input formats of the rxncon language: SBtab [186] spread sheet in Microsoft Excel format and Quick format. The spread sheet follows the SBtab standards and consists of four separated sheets, two for defining the mechanistic information of the reconstructed network and two sheets for defining the different reaction classes. The reconstructed information is stored into the rxncon reaction and the contingency sheet. The reaction sheet has three columns for each specification, which correspond to components within the sheet: one for the component name (e.g. !ComponentA:Name), one for the domain (e.g. !ComponentA:Domain) and one for the residue (e.g. !ComponentA:Residue). Additionally, there is a column for the definition of the reaction type (!Reaction). The reaction type within this column must refer to a unique key in the ReactionTypeDefinition sheet (!UID:ReactionKey, see below). There is another column for annotating literature sources (!Literature:Identifiers:pubmed), e.g. PubMed identifier and two additional columns for the quality of the empirical evidence of the reaction (type of the experiment) or the confidence in the results (!Quality) and comments

Table S4: Description of some proteins involved in the high-osmolarity-glycerol pathway in yeast

Proteins	Description
Sln1,	Transmembrane histidine phosphotransfer kinase and osmosensor
Msb2, Hkr1	mucin-like transmembrane sensors
Sho1	transmembrane protein involved in osmosensing and other signalling pathways, scaffold for Pbs2, Ste11 and other
Cdc42	Small Rho GTPase; binds Ste20 and activates the kinase domain of Ste20
Ste20	PAK (p21-activated protein kinase); activated by Cdc42; autophosphorylation
Ste11	MEKK (MEK kinase); activated by Ste20
Ypd1, Ssk1	Together with Sln1 phosphorylation system, phosphorylated Ssk1 cannot activate the downstream MAP kinase cascade
Ssk2, Ssk22	MEKK (MEK kinase); activated by Ssk1
Pbs2	MEK (MAPK/ERK kinase); scaffold protein for the MAPK cascade; activated by Ssk2 and/or Ssk22; activated by Ste11
Hog1	MAP kinases; activated by Pbs2
Hot1	Transcription factor for glycerol biosynthetic genes; activated by Hog1

or thoughts to the respective rxncon statement (!Comment). The contingency sheet has three columns: one for the Target (!Target), one for the Contingency type (!Contingency) and one for the Effector (!Modifier). The column !Target contains reaction terms that are regulated, defined by the unique identifier from the reaction sheet (!UID:Reaction) and is also used to define Boolean statements. The Effectors of a contingency term are described in the column !Modifier. The !Contingency column contains the contingency type, defining the influence of an Effector (!Modifier) on a reaction term (!Target) (see Table). As in the rxncon reaction sheet there are three additional columns available (!Quality, !Reference:Identifiers:pubmed, !Comment) to add additional information for a rxncon statement.

The different reactions types, which can be used within the reconstruction, are defined within the ReactionTypeDefinition sheet. Each reaction type needs a unique identifier (!UID:ReactionKey) as well as a definition of its reactants in respect of their type (Protein, mRNA, Gene or Any (!MolType)) and their required resolution (component-level, domain-level, residue-level (!Resolution)). The semantic of a reaction type is defined by its skeleton rule and is described in the column !SkeletonRule. To create a new reaction type or elemental modification state it is sufficient to add a new entry to the respective list. All elemental modification states used within

the ReactionTypeDefinition have to be defined in the ModificationTypeDefinition sheet. A new modification property has to have a unique identifier (!UID:ModificationType) and a unique label (!UID:ModificationLabel) [122]. A template is shown in Supplementary File SF13.
The Quick format is a text based format and can be written in any editor. It follows the syntax presented in Chapter 2.

```
A_[b]_ppi+_B_[a]; !  A_[(r)]-p
C_p+_A_[(r)]
```

The ';' separates the rxncon reaction from its contingencies. Within this format only predefined rxncon reaction types (Supplementary Table S5) can be used.

Table S5: Table of the dependencies for the rxncon library

!UID:Reaction	!UID:ReactionKey
phosphorylation	p+
dephosphorylation	p-
auto-phosphorylation	ap+
phosphotransfer	pt
guanine-nucleotide-exchange	gef
GTPase-activation	gap
ubiquitination	ub+
truncation	cut
protein-protein-interaction	ppi
intra-protein-interaction	ipi
interaction	i
protein-gene-interaction	bind
transcription	trsc
translation	trsl
synthesis	syn
degradation	deg

Python library

rxncon was implemented using the Python programming language[1] version 3.5. and tested under Unix, OSX and Microsoft Windows. For Windows, we recommend Anaconda[2] to install python. Pip is used to support the distribution of the package and to install the rxncon packages from Python Package Index (PyPI). It can be installed by typing 'pip install rxncon'. rxncon has some dependencies listed in Table S6. There are three possibilities to install pyeda under Windows: 1) installing pyeda from scratch 2) to download a precompiled version of pyeda[3] and installing it manually 3) typing 'pip install rxncon'. We will automatically install pyeda, using a precompiled pyeda version from our server (recommended). We support pyeda for python 3.5 and 3.6. The source code is available on gitHub[4].

Table S6: Table of the dependencies for the rxncon library

Packages	Description
pytest	Framework for testing applications
numpy	Package for scientific computing
scipy	Package containing scientific tools
click	Package for command line interface
click-log	Logging integration for click
colorama	Package for colouring terminal text
xlrd	Package for reading Excel files
networkx	Package for creation and analysis of networks

Scripts

We provide command line scripts, which are automatically installed by installing rxncon via pip. The scripts will automatically be available globally under Unix and OSX but not under Windows. If you are using Anaconda you can find them within the Anaconda Scripts folder. The scripts are listed in Table S7. The expected input of each script can vary please use the -help option for further help.

[1] https://www.python.org/
[2] https://www.continuum.io/downloads
[3] http://www.lfd.uci.edu/~gohlke/pythonlibs/#pyeda
[4] https://github.com/rxncon/rxncon

Table S7: Table of the different scripts provided by the rxncon framework.

Script	Description
rxncon2regulatorygraph.py	Translates the rxncon system into a xgmml format containing the information for a regulatory graph
rxncon2reactiongraph.py	Translates the rxncon system into a xgmml format containing the information for a reaction graph
rxncon2srgraph.py	Translates the rxncon system into a xgmml format containing the information for a species-reaction graph
rxncon2boolnet.py	Translates the rxncon system into a BoolNet file
rxncon2bngl.py	Translates the rxncon system into a BNGL file

Model simulation

bipartite Boolean model simulation

All simulations were carried out using the R-Package BoolNet (v 2.1.3) [160]. The rxncon system can be exported into a BoolNet compatible file (*.boolnet). Within the BoolNet model state nodes and reaction nodes are referred to as state targets and reaction targets. To fulfil the naming restrictions of BoolNet and to keep the names of the nodes identifiable and unambiguous we introduce abbreviations for the names of reaction and state nodes. Reaction nodes are abbreviated with R and the state nodes are abbreviated with S followed by a consecutive numbering. Additionally, two csv files are provided: one file (ending with *_symbols.csv) mapping an abbreviation to its respective node and another file (ending with *_initial_vals.csv) containing the initial values of all nodes of the system. Note that the system is initialised per default with neutral state nodes and generic component nodes as True and all other nodes as False.

For the simulation process it is expected that the output is responsive to the input (sign of the dependence does not matter) and that the simulation starts with input either True or False. The simulation runs until an attractor is reached. If the attractor is a point attractor, the system state of this point attractor is used as starting point for the next simulation round but the logical value of the input signal is inverted. The simulation runs until we reach another attractor. If the output did not switch on in the first round, an active output should be observed now. This procedure can be repeated iteratively (invert logical value of input state and simulate until an attractor reached) until an attractor is reached that was already observed.

Agent based simulation: NFsim and BioNetGen

For a ODE simulation with BioNetGen the following commands can be used at the end of a BNGL file:

```
generate_network({overwrite=>1});
simulate_ode({suffix=>"ode_before",t_end=>50,n_steps=>500});
setConcentration("insulin(IRD)", "1000");
simulate_ode({suffix=>"ode_during",t_end=>50,n_steps=>500});
setParameter("ins_deg_0", "100");
simulate_ode({suffix=>"ode_after",t_end=>50,n_steps=>500});
```

`generate_network` will calculate the complete reaction network of the system and write it into a `.net` file. The attribute `overwrite=>1` will overwrite this file if the network is generated again. `simulation_ode` will infer an ODE system of the network and calculates the average concentration of the molecules during the simulation. The results for the observables will be written into a `.gdat` file and the simulation results for all species will be written into a `.cdat` file. The attribute `suffix=>` will set the suffix of the files the simulation result are stored and the attributes `t_end=>` and `n_steps=>` defines the simulation time and the sampling number during the simulation respectively. Within BioNetGen the command `setConcentration` can be used to adjust molecule concentrations during the simulation. To do so, the molecule has to be specified in terms of states and sites. The `setParameter` command can be used to change the initial value of a parameter, a large change compared to the previous value, e.g. for the degradation of a molecule make sure that the molecule almost instantaneously disappears.

NFsim simulation

For a NFsim simulation a XML file has to be generated using BioNetGen (v 2.2.6) that can be read by NFsim. Therefore the BNGL file has to end with `writeXML();`. For the simulation we use a Run NF script (Supplementary File SF12)

Supplementary Figures

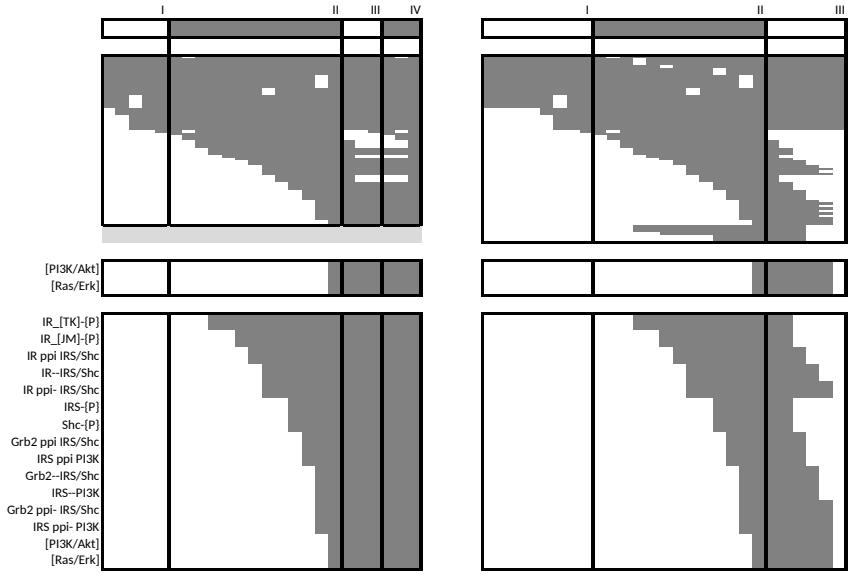


Figure S1: **Simulation of the exported insulin bipartite Boolean model.** A) Simulation without dephosphorylation reaction nodes. The heatmap of the simulations show the synchronous update for each reaction node and state node (rows) over time (columns). The colours indicate the Boolean state of the nodes: grey = True, white = False. The simulation is initiated with the default configuration, but in absence of insulin. The model is simulated until the first attractor is reached. The outputs remains off. All following simulations use an adapted version of the attractor from the previous simulation. Next, we set the neutral state of insulin to True, and repeated the simulation until the next attractor is reached. The outputs turn on. To remove insulin from the system we set all insulin state nodes as well as all reaction nodes, producing insulin, to False. We run the simulation until we reached the next attractor. The output signals do not turn off. Since the current attractor was not seen before, we run a new simulation starting from the last attractor, but with insulin True, until we reached the next attractor. This attractor is identical to the second attractor. B) Simulation with dephosphorylation reactions. The simulation is done the same way as for panel A. The third attractor is equal to the first attractor. The output signals respond to the input signal as expected.